AD-A079 951 NAVAL RESEARCH LAB WASHINGTON DC F/G 9/2
MULTIPLE PLATFORM SENSOR INTEGRATION MODEL: MULSIM COMPUTER PRO--ETC(U)
DEC 79 A GRINDLAY
UNCLASSIFIED NRL-8358 SBIE-AD-E000 349 NL

1 of 2

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 14 NRL [redacted] 8358 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9
4. TITLE (and Subtitle) 6 MULTIPLE PLATFORM SENSOR INTEGRATION MODEL: MULSIM COMPUTER PROGRAM		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) 10 A. Grindlay		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory ✓ Washington, D.C. 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NRL Problem R12-18.801 62712N SF12-133-401
11. CONTROLLING OFFICE NAME AND ADDRESS Department of the Navy Naval Sea Systems Command (NSEA-0321) Washington, DC 20362	11 31	12. REPORT DATE [redacted], 1979
		13. NUMBER OF PAGES 105 12
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 16 F12133		15. SECURITY CLASS. (of this report) UNCLASSIFIED 106
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17 SF12133401

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18 SBIE

18. SUPPLEMENTARY NOTES

19 AD-E111 347

Accession For
NTIS GRA&I
DDC TAB
Unannounced
Justification
By
Distribution/
Availability Codes

Dist.	Avail and/or special
A	

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Multiple platform
Sensor integration
Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Multiple Platform Sensor Integration Model (MULSIM) simulates the operation of a track management system that is receiving inputs from a large number of widely distributed platform/sensors. The model consists of two basic parts: a stimulator and a track correlation/association module. The stimulator produces noisy detections from an input scenario. The detections from each platform are subjected to a correlation/association process and the detections which associate with existing tracks are integrated with selected detections from other platforms to produce updated position estimates of system tracks.

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

i

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251 950

CONTENTS

MULTIPLE PLATFORM SENSOR INTEGRATION MODEL: MULSIM COMPUTER PROGRAM

1. INTRODUCTION

1.1 Background

The Navy has always been interested in developing the means to effectively integrate the activities of individual units engaged in multiunit operations. Efforts in this direction led to the development of the Navy Tactical Data System (NTDS), which is basically a computer-aided manual system. Since its development in the 1950s there have been major technological improvements. These developments, together with improved communications systems such as the Joint Tactical Information Distribution System (JTIDS), and navigation satellites such as NAVSTAR have brought the automatic, multiple-platform sensor integration system into the realm of possibility.

The system that has been modeled consists of two or more ships, each having one or more radar/ESM systems on board (see Fig. 1.1). The surveillance systems are assumed to be detecting targets either on an individual basis or jointly with other surveillance systems. A communication system with capabilities similar to JTIDS and a navigation system capable of giving accurate fixes on all participating platforms are assumed to exist. The combined system has the ability to transmit and assimilate data and provide smoothed tracking information to all participating platforms.

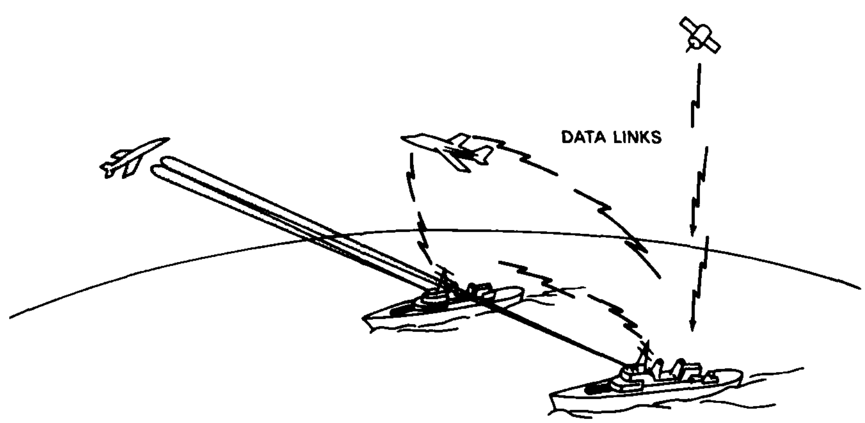

Fig. 1.1 — Mult·ple Platform Sensor Integration System

Manuscript submitted August 14, 1979.

From a global point of view, the most obvious benefit to be derived from the development of this system is the presentation of the overall tactical environment to Fleet commanders. System survivability is an important feature. Platforms will still be able to operate with tracking information from other platforms in the event that their surveillance systems become inoperable or are shut down in EMCON situations. The effects of stand-off main-lobe jammers can also be minimized as shown in Fig. 1.2. More fundamental, however are the benefits to be derived in track management. Improved tracking performance can be expected from frequent updates that occur when several sensors are producing detections and from increased accuracy produced by the cross hairing of targets (see Fig. 1.3).

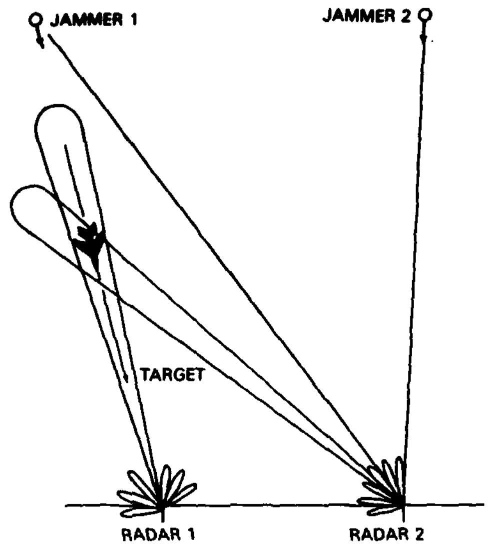

Fig. 1.2 — Stand-off main-lobe jammers

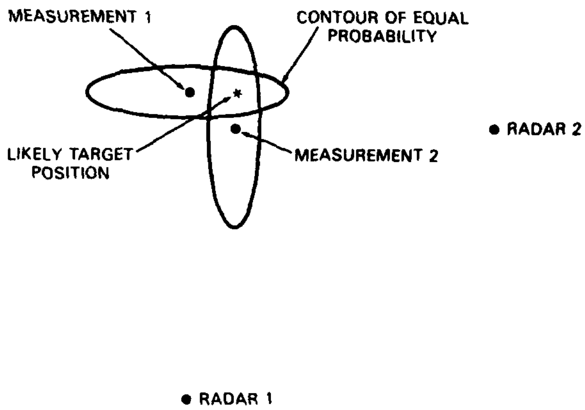

Fig. 1.3 — Effects of cross-hairing

There are two reasons for developing this model. The primary consideration is the development of a system architecture,* i.e., actual development of algorithms and techniques for track correlation/association, track management, and updating of tracks. This architecture is hinged upon the concept of obtaining the best target information from sensors while using the smallest amount of channel capacity. The second consideration is having the capability of examining the performance of the system, in particular the propagation of errors through the system.

To reiterate, the model will serve as a foundation for future software development and at the same time allow the user to demonstrate the advantages/limitations inherent in a multiple platform sensor integration system.

1.2 Model Architecture

The model consists of two basic parts: a stimulator and a track correlation/integration system (Fig. 1.4). The stimulator takes initial target positions, headings, and velocities from an input scenario and determines their position at some later time designated by a radar sector crossing. It then adds measurement errors to the true coordinates of the radar detections and inputs them to the track correlation/integration module. The detections from each individual platform are subjected to a correlation/association process, and the detections that associate with existing tracks are integrated with selected detections from the other platforms to produce updated positions for the system tracks.† The stimulator and the track correlation/integration system are controlled and linked by the executive routine. The executive routine also controls the initiation process and the loading of track files.

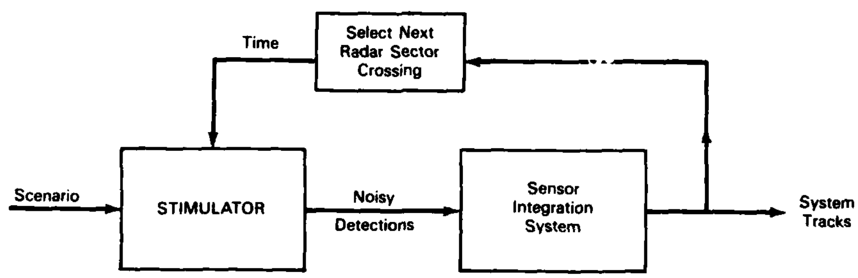

Fig. 1.4 — Basic modules

*See Refs. 1 and 2 for a more detailed discussion of system architecture and operating philosophy.
†See Refs. 1 and 4 for sample tracking outputs from MULSIM.

2.0 MULSIM PROGRAM

2.1 Executive Module

The executive module has four basic functions: (a) to read the program inputs and initialize the scenario, (b) to load the system track files, (c) to schedule events and, (d) to call the various subroutines in a logical sequence that simulates the functional flow of an operating system. These functions are handled by subroutines INITAL, LOAD, and NEXRAD and the executive routine, respectively.

2.1.1 *MULSIM Executive Routine*

The executive routine drives the MULSIM program. Besides calling for the initialization of the scenario and the loading of the track files, the executive routine calls each subroutine, in a logical sequence, whenever the NEXRAD subroutine schedules a sector-crossing event. This procedure can best be followed by referring to Fig. 2.1.

The process is started by setting the game time equal to zero. Subroutine INITAL is then called to set the initial values of scenario parameters. As the program is currently configured there is no formal input/output (I/O) structure. Initial positions of targets and ships and parameters relating to trajectories are set in INITAL on a card-by-card basis, i.e., there are no formatted inputs.

The next step in the process is the loading of the sector track files and the initialization of the tracking filter. The positions of all the targets and ships with respect to each ship are determined at time = 0 s and at time = 1 s in each ship's stabilized coordinate system and deck-plane coordinate system. The subroutines called for this purpose are TRKGEN, SHPGEN, MOTION, SCOORD, TCOORD and STAB1. Each subroutine is described in detail in the track correlation/integration section.

The LOAD subroutine uses the generated position information to establish target velocities and initialize the covariance matrices for the Kalman filter. LOAD also uses the position information to load the sector track files. The area around each ship is divided into 64 angular sectors, and target tracks are assigned to sector track files according to their current location.

When the loading and initialization process has been completed, the program starts to cycle through the main loop of the routine. The program exercises this loop each time a radar makes a sector crossing. The time at which a sector crossing takes place and the sector number of the sector the radar has just crossed is determined by subroutine NEXRAD. If there are targets in the sector designated by NEXRAD or in adjacent sectors, the next step is to update the position of all targets and ships to the sector crossing time. Subroutines TRKGEN and SHPGEN are called to give the updated latitude and longitude of all the targets and ships, and the ship's motion is accounted for by calling subroutine MOTION, which provides the current pitch and roll of each ship. The new coordinates of all the targets and ships in each ship's stabilized coordinate system are found by calling subroutines TCOORD and SCOORD.

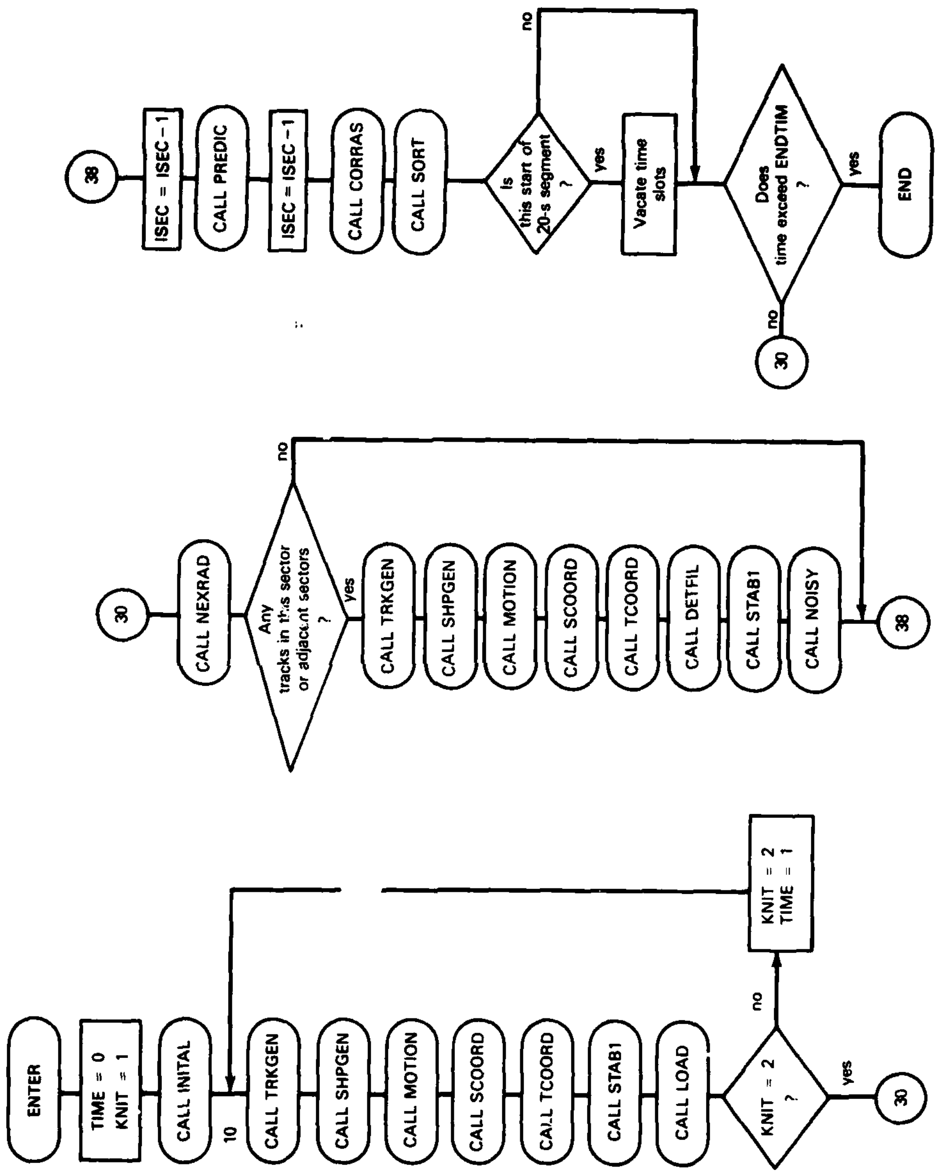

Fig. 2.1 — MULSIM Executive routine

The targets located in the sector designated by NEXRAD are identified and their numbers are loaded into detection files. Each sector is further divided into range bins (200 currently) and, corresponding to each range bin, there is an existing detection file. The loading of these detection files takes place in subroutine DETFIL.

Before the correlation/integration process can be attempted, noise must be added to the "true" coordinates of the targets to approximate the measurement process. This is accomplished by first transforming the stabilized coordinates to deck-plane coordinates* in subroutine STAB1 and then injecting noise by selecting samples from a normal noise distribution derived from a random number generator. This function is performed by subroutine NOISY, which also transforms the noisy deck plane coordinates back to the stabilized coordinate system. All of the correlation/integration process is carried out in the stabilized coordinate system.

There is one additional bookkeeping function performed prior to the correlation/integration process. The system sector track files must be kept current. Each sector has a file that contains all of the tracks currently located in that sector. The PREDIC subroutine is called to update these track files to predict the position of the tracks at the sector crossing time. To account for processing delays the PREDIC subroutine is applied, not to the tracks in the sector designated by NEXRAD, but to the previous sector. An additional time delay is introduced after PREDIC is called by stepping back one more sector before starting the correlation/integration process.

The correlation process is started by calling subroutine CORRAS. The tracks located in the sector under consideration are individually selected for correlation with detections. The detections located in the nine range/sector bins contiguous to the track are said to be correlated with the track, and the statistical distance between the track and each detection is calculated.

The association process is concerned with the resolution of conflicts that might arise in the correlation process, and this is also handled in the CORRAS subroutine. Conflicts occur when two or more tracks are correlated with the same detection. If this is the case, statistical distances are compared and the detection is declared to be associated with the track having the smallest statistical distance to the detection.

The next step is to sort the associated detections. SORT subroutine places detections in three categories: those associated with participating platforms; those associated with tracks that one's own ship is responsible for updating; and those associated with tracks which ownship is not responsible for updating.

Although it is not currently being done, it is planned eventually to use those detections in the first category for reducing bias errors. Positional information from detections in the second category is stored for the updating process, and if the detection belongs to the third category, SORT calls subroutine TIMCON to determine if a time slot is available for transmitting data over the link.

---

*See Ref. 2 for description of coordinate systems.

The executive routine is also concerned with the flagging of time slots. The status of the 1-s time slots over a 1-min period is taken into account. This 1-min period is divided into segments of 20 s and the executive routine is responsible for setting the flags associated with each 1-s time slot in the segment. At 20-s intervals the executive routine sets the flags in the next 20-s segment to 0. For each time slot this indicates that no data have been sent over the link during that 1-s period. As data are transmitted the corresponding flags are set equal to 1.

The LNKDET subroutine is called to start the updating process. For each track the LNKDET subroutine merges the detections from the communications link with those from ownship in a sequential file.

2.1.2 *Subroutine INITAL*

As the program is currently configured there is no formal input/output structure. Consequently INITAL is used to define the scenarios, set parameters, initialize arrays, and define constants. Table 2.1 defines the arrays, parameters, constants, etc. which are set in INITAL. Variables are listed in order of appearance.

Table 2.1 — Functions Performed in INITAL

Fortran Variable	Description
RAD	Conversion factor, radians to degrees
DIM1, DIM2, DIM3	Dimensions for setting size of arrays in KALMAN
TIMLAG	Time lag used in UPDATE
LASDET	Location of last available space in file that is loaded in DETLOC
NEXDET	Location of next available space in file that is loaded in DETLOC
RNGDIM(I,J)	Dimension of range bin in meters for radar I on platform J
N(I)	Standard deviation of noise in measurement of targets and position of platforms. Used to determine the measurement covariance matrix with respect to platform J when measurement is made at platform I.
N2(I)	Standard deviations of noise in measurements, used to determine covariance matrix with respect to platform I's stabilized coordinate system
SIGAZD(I,J)	Standard deviation of azimuth measurement noise for radar J on platform I
SIGELD(I,J)	Standard deviation of elevation measurement noise for radar J on platform I

Table 2.1 (Continued) — Functions Performed in INITAL

Fortran Variable	Description
RHOD(I,J)	Standard deviation of range measurement noise for radar J on platform I
LISDET(I)	Linkage device used in DETLOC to reserve and vacate locations in files
DETSC(I)	Pointing device use in CORRAS to pinpoint location of last entry to file
FILEX(I,J)	Pointing device used in SORT to pinpoint location of last entry to file
FILID(I)	Linkage device used in SORT to link all locations in a file that are associated with a particular track
NS	Number of platforms in scenario
AZINT(I,J)	Initial azimuth of radar I on platform J (deg)
RVEL(I,J)	Rotation rate of radar I on platform J (deg/s)
SILAT(I)	Initial latitude of platform I (deg)
SILOG(I)	Initial longitude of platform I (deg)
SIHT(I)	Initial height of platform
SVEL(I)	Velocity of platform I (m/s)
NR(I)	Number of radars on platform
NT	Number of targets in scenario
SECTIM(I,J)	Time required by radar I on platform J to sweep over one angular sector
TILAT(I)	Initial latitude of target I (deg)
TILOG(I)	Initial longitude of target I (deg)
TIHT(I)	Initial height of target I (in.)
SIHD(I)	Initial heading of platform I (deg)
TIHD(I)	Initial heading of target I (deg)
TVEL(I)	Velocity of target I (m/s)
ER	Equatorial radius of the earth (m)
PR	Polar radius of the earth (m)
TIV(I)	Angular velocity of target I on great circle route (rad/s)
SIV(I)	Angular velocity of platform on great circle route (rad/s)
RMAG(I)	Roll magnitude of platform I (rad)
PMAG(I)	Pitch magnitude of platform I (rad)

Table 2.1 (Concluded) — Functions Performed in INITAL

Fortran Variable	Description
WOR(I)	Roll rate of platform I (rad/s)
WOP(I)	Pitch rate of platform I (rad/s)
RPHASE(I)	Initial roll phase angle for platform I (rad)
PPHASE(I)	Initial pitch phase angle for platform I (rad)
LASTM(I)	Last available location in file used in MPTFIL
DPOPM(I)	Indicator used in MPTFIL
FULLM(I)	Number of available locations in file used in MPTFIL
NEXTM(I)	Next available location in file used in MPTFIL
LISTM(I,J)	Linking device used in MPTFIL
DROPD(I)	Indicator used in DUMFIL
LASTD(I)	Last available location in file used in DUMFIL
FULLD(I)	Number of available spaces in file used in DUMFIL
NEXTD(I)	Next available space in file used in DUMFIL
LISTD(I,J)	Linking device used in DUMFIL
FULLNK	Number of available spaces in file in LNKLOC
LASLNK	Last available space in file used in LNKLOC
NEXLNK	Next available space in file used in LNKLOC
LISLNK(I)	Linking device used in LNKLOC
G(I,J)	Array used in state equation in KALMAN
H(I,J)	Array used in observation equation in KALMAN

2.1.3 *Subroutine LOAD*

Subroutine LOAD is called twice by the MULSIM executive routine. Once at time = 0 s and once at time = 1 s. Positions of all the targets and platforms with respect to every other platform are determined at time = 0 s. This calculation is carried out with the true target/platform locations, and the positions are determined in each platform's stabilized coordinate system. The position coordinates are saved for the next pass through LOAD, and control is returned to the executive routine. On the second pass through LOAD, positions are again determined at time = 1 s and velocity estimates are made from the position changes over the 1-s time interval. This information is used to load the estimated state vector for the tracking filter.

LOAD next calls the STAB2 subroutine for noisy deck-plane position coordinates that are used in subroutine COVOWN to determine initial values of the measurement covariance matrix. These values are used in turn to load the covariance matrix that corresponds to the state vector estimate. In lieu of a track-initiation process, LOAD is also used to assign MPT and dummy track numbers and to load the sector files. It is also used to initialize the NEXSEC, TIMNEX, and TLAST arrays. The NEXSEC and TIMNEX arrays contain the number of the sector that each radar will next cross and the time at which this sector crossing will take place. The TLAST array contains the time at which each track was last updated. Initially all the elements of TLAST are set equal to 1 s.

The entire process is outlined by a macro flowchart in Fig. 2.2.

2.1.4 *Subroutine NEXRAD*

Subroutine NEXRAD is called by the Program MAIN. Its primary function is to determine which radar on which platform will next make a sector crossing and then record the time at which this event will take place.

The area surrounding each ship is divided into 64 angular sectors. The sectors are numbered clockwise from true North with the first sector to the right of North being assigned the number 1.

The flow of logic through the subroutine is outlined in Fig. 2.3. The TIMNEX(I,J) array contains the time at which radar I on platform J will next make a sector crossing. NEXRAD interrogates this file for the lowest time. This determines which radar will be the first to make a sector crossing. The sector number of the sector that radar I is currently scanning is stored in the NEXSEC(I,J) file. This number is incremented when radar I on platform J has the lowest value in the TIMNEX file. The TMRK(ISEC,I,J) file is also updated in NEXRAD. This file records the time at which radar I on ship J crossed from sector ISEC to ISEC+1. Before leaving NEXRAD, TIMNEX(I,J) is also increased by the time required by radar I to sweep across a sector.

2.2 SYSTEM STIMULATOR

2.2.1 *Subroutine MOTION*

Subroutine MOTION is called by the executive routine (MAIN). For a given time, MOTION calculates the current roll and pitch of each platform. The roll and pitch of each platform are assumed to be time-varying sinusoidal functions with specified initial values, magnitudes, and frequencies. Table 2.2 defines the variables used in MOTION.

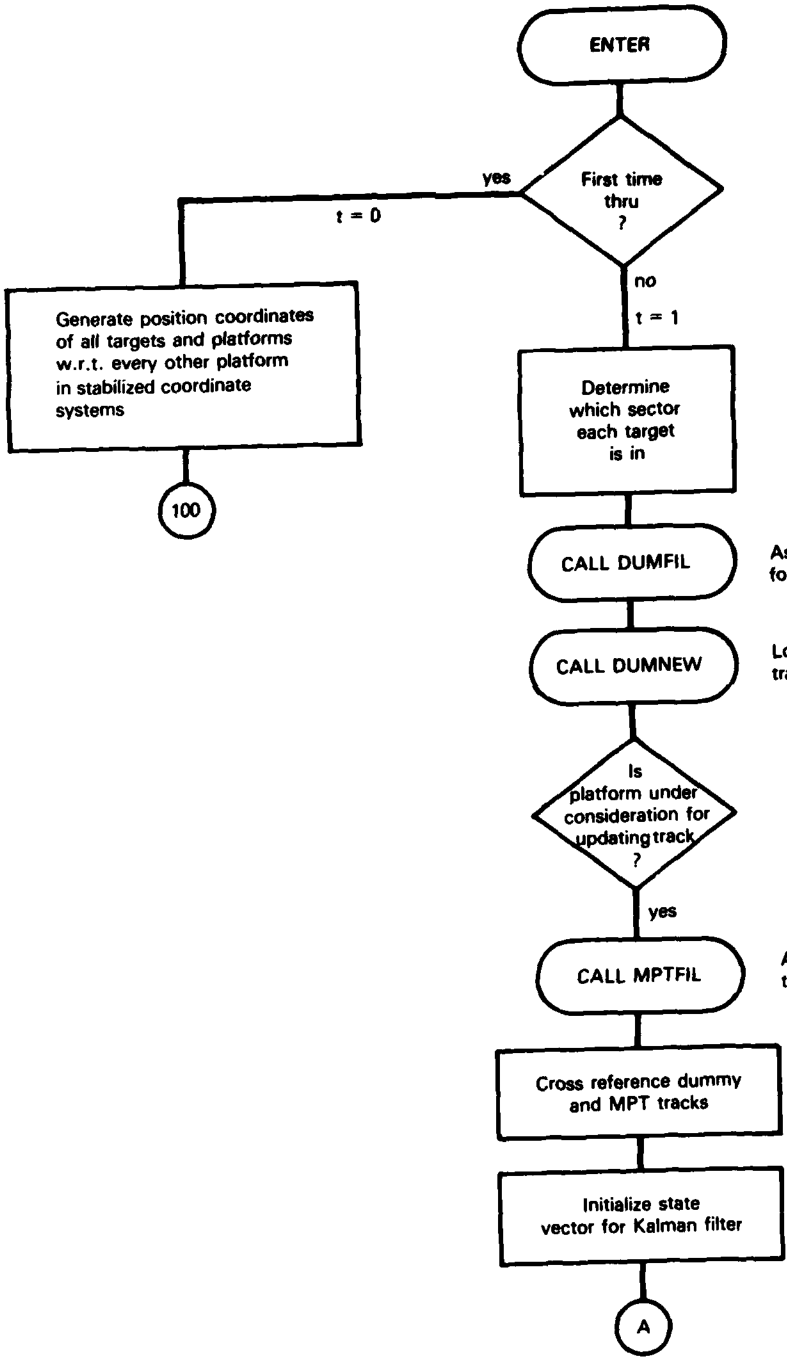

Fig. 2.2 — Subroutine LOAD macro flowchart

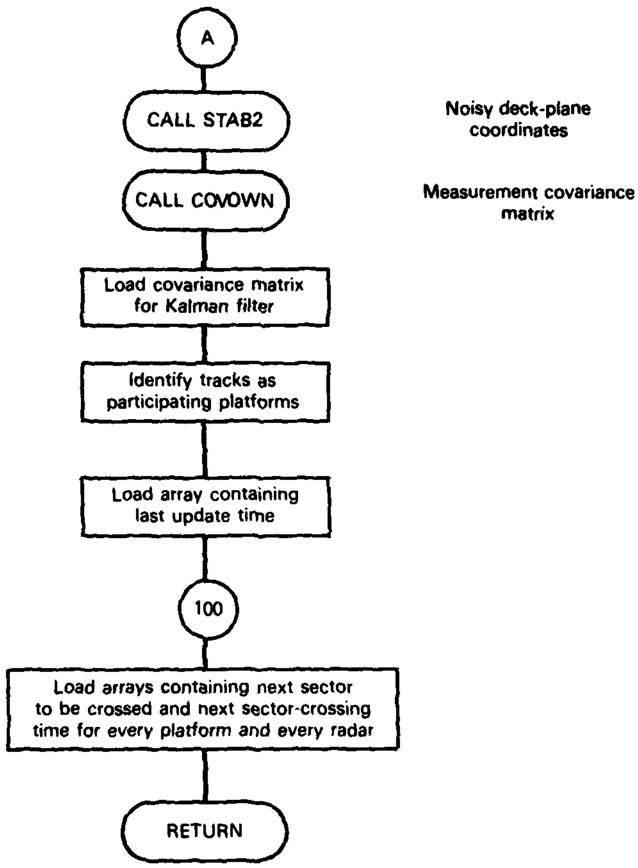

Fig. 2.2 (Concluded) — Subroutine LOAD macro flowchart

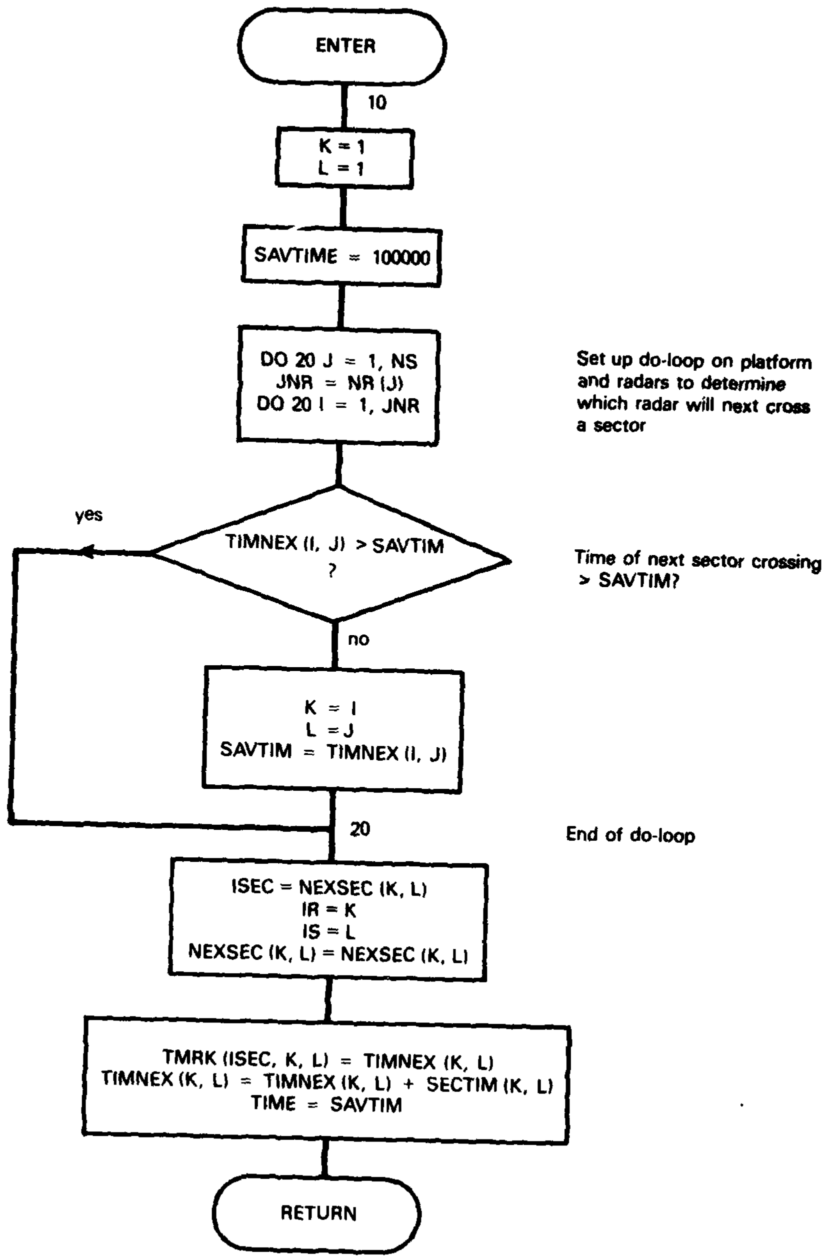

Fig. 2.3 — Subroutine NEXRAD

Table 2.2 — Variables in Subroutine MOTION

Fortran Variable	Description
TIME	Time (s)
NS	Number of platforms in scenario
ARG	Roll or pitch for unit magnitude
WOR(I)	Angular roll frequency for platform I (rad/s)
RPHASE(I)	Initial roll for ship I (rad)
ROLL(I)	Current roll position for platform I (deg)
RMAG(I)	Magnitude of roll for platform I (deg)
WOP(I)	Angular pitch frequency for platform I (rad/s
PPHASE(I)	Initial pitch for platform I (rad)
PITCH(I)	Current pitch position for platform I (deg)

2.2.2 *Subroutine SHPGEN*

Subroutine SHPGEN determines the current latitude, longitude, and heading of all platforms in the scenario from current positions and velocities. As the program is currently configured, the platforms are confined to moving on great circle routes over an oblate spheroid at constant speed. The platforms' angular velocity, initial heading, latitude, and longitude are provided by INITAL. This reduces the problem to a simple exercise in spherical trigonometry. Almost 50% of the logic in the subroutine is concerned with resolving problems encountered at trigonometric discontinuities (± 90°, 180°, 360° etc.). Table 2.3 defines the variables used in SHPGEN. They are listed in the order of their appearance in the program listing.

Table 2.3 — Variables in SHPGEN

Fortran Variable	Description
SIV(I)	Angular velocity of platform I on great circle route (rad/s)
TIME	Time (s)
C	Central angle transcribed by platform under consideration
YP	Sine of central angle
ZP	Cosine of central angle
SIHD(I)	Initial heading of platform I measured from true North (deg)

Table 2.3 (Concluded) — Variables in SHPGEN

Fortran Variable	Description
SILAT(I)	Initial latitude of platform I (deg)
SILOG(I)	Initial longitude of platform I (deg)
XG,YG,ZG	Direction cosines of platform's current position in geocentric coordinate system
SLAT(I)	Current latitude of platform I (deg)
SLOG(I)	Current longitude of platform I (deg)
SIHT(I)	Initial height of platform I (m)
SHT(I)	Current height of platform I (m)
SHD(I)	Current heading of platform I (deg)

2.2.3 *Subroutine TRKGEN*

Subroutine TRKGEN is very similar to subroutine SHPGEN. TRKGEN performs the same function for targets that SHPGEN performs for platforms. The current latitude, longitude, and heading are determined for all targets in the scenario. The targets are confined to moving on great circle routes at constant altitude and speed over an oblate spheroid. The targets' angular velocity, initial heading, latitude, and longitude are provided by INITAL. Table 2.4 defines the variables used in TRKGEN, listed in the order of their appearance in the program listing.

Table 2.4 — Variables in TRKGEN Subroutine

Fortran Variable	Description
TIV(I)	Angular velocity of target I on great circle route (rad/s)
TIME	Time (s)
C	Central angle transcribed by target under consideration
YP	Sine of central angle
ZP	Cosine of central angle
TIHD(I)	Initial heading of target I measured from true north (deg)
TILAT(I)	Initial latitude of target I (deg)
TILOG(I)	Initial longitude of target I (deg)

Table 2.4 (Concluded) — Variables in TRKGEN Subroutine

Fortran Variable	Description
XG, YG, ZG	Direction cosines of targets' current position in geocentric coordinate system
TLAT(I)	Current latitude of target I (deg)
TLOG(I)	Current longitude of target I (deg)
THT(I)	Current height of target I (m)
THIT(I)	Initial height of target I (m)
THD(I)	Current heading of target I (deg)

2.2.4 *Subroutine SCOORD*

The current range, azimuth, and elevation of all the platforms with respect to a specified platforms' stabilized coordinate system is determined by subroutine SCOORD. The locally stabilized coordinate systems are centered at each platform's c.g. with the z-axis pointed upward along the local gravity vector, the y-axis pointed toward true North, and the x-axis lying due east (see Fig. 2.4). Azimuth is measured clockwise from the y-axis.

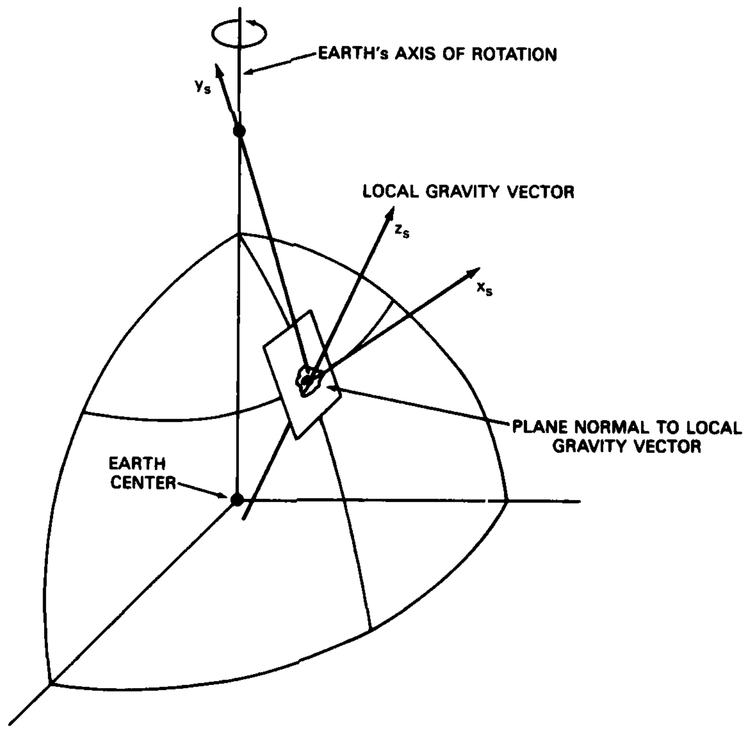

Fig. 2.4 — Locally stabilized coordinates

If the latitude, longitude, and altitude of each platform are known, it is a simple exercise in spherical trigonometry to determine the stabilized coordinates of each platform in some other platform's stabilized coordinate system and hence the respective range, azimuth, and elevation. The variables used in this process are listed in Table 2.5 in the order of their appearance in the program listing.

Table 2.5 — Variables in SCOORD

Fortran Variable	Description
NS	Number of platforms in scenario
NT	Number of targets in scenario
ISHIP	Platform under consideration
ER	Equatorial radius (m)
PR	Polar radius (m)
RAD	Conversion factor, degrees to radians
SLAT(I)	Current latitude of platform I (deg)
RHOT	Local Earth radius at platform I (m)
RHOS	Local Earth radius at platform ISHIP (m)
SLOG(I)	Current longitude of platform (deg)
X,Y,Z	Geocentric coordinates of platform I
XP, YP, ZP	Stabilized coordinates of platform I in ISHIP's stabilized coordinate system
AZ(K,J)	Azimuth of platform K with respect to platform J (deg)
EL(K,J)	Elevation of platform K with respect to platform J (deg)
RG(K,J)	Range of platform K with respect to platform J (m)
ISEC	Sector containing platform K in platform J's coordinate system
KSEC(ISEC,J)	Indicator which indicates that platform J has a target in sector ISEC
SHT(J)	Altitude of platform J (m)

2.2.5 *Subroutine TCOORD*

Subroutine TCOORD essentially parallels subroutine SCOORD; i.e., the range, azimuth, and elevation of all targets in the scenario are determined in a specified platform's stabilized coordinate system. The variables not used in subroutine SCOORD are given in Table 2.6.

Table 2.6 — Variables Not Used in SCOORD

Fortran Variable	Description
TLAT(I)	Current latitude of target I (deg)
TLOG(I)	Current longitude of target I (deg)
THT(I)	Current altitude of target I (m)

2.2.6 *Subroutine DETFIL*

Subroutine DETFIL is called by the executive routine to assign detection numbers to targets and load them in their respective range bin files. Technically speaking, DETFIL is *not part of the stimulation process*. In an operating system the DETFIL function would be performed on noisy measurements made by the system; however, in MULSIM the DETFIL function is performed on the stabilized true target positions. This was done to eliminate problems associated with targets flying along sector lines and hopping from one sector to the other as noise was injected into their measurements. Since the DETFIL subroutine performs its operations before the completion of the stimulation process, it has been included in the system stimulator section.

Subroutine DETFIL is called for a specified radar, platform, and sector. The first step in the process is to determine the angular limits of the specified sector and then run through all the platforms and targets in the scenario to see if they lie within this sector. The next step is to determine which range bin contains the target and to load the assigned detection numbers into a linked file that contains all the detection numbers assigned to each individual range bin. Aside from determining whether the target under consideration has moved into the sector during the sector crossing time and zeroing out an array used in the correlation subroutine, this essentially completes the process. Table 2.7 describes the FORTRAN variables used, and Fig. 2.5 is a flowchart of the subroutine logic.

Table 2.7 — Variables in DETFIL

Fortran Variable	Description
IR, IS, ISEC	Identification numbers of radar, ship, and sector under consideration
NT, NS	Number of targets and platforms in scenario
LSTBIN(I,J,K,L)	Array containing identification number of last target to be placed in linked file for sector I, range bin J, radar K, and platform L
AZLO	Lower boundary of sector ISEC (deg)
AZHJ	Upper boundary of sector ISEC (deg)

Table 2.7 (Concluded) — Variables in DETFIL

Fortran Variable	Description
AZ(I,J), RG(I,J)	Azimuth, range of target/platform I w.r.t. platform J (deg)
JRN	Range bin identification number
IDET(IR,IS)	File containing next-detection identification number to be assigned by radar IR on platform IS
RNGDIM(IS,IR)	Range dimension of range bins for radar IR on platform IS
IDTA(ID, IR, IS)	File which links detections and targets. File contains target/platform number that corresponds to detection ID from radar IR on platform IS.
ITAG(I,IR,IS,ISEC)	Indication that target I was assigned detection number in sector ISEC
LNKBIN(ID,IR,IS)	Linking device that links all the detections from a particular range bin for radar IR on platform IS
TRATG(ID)	Flag that indicates that detection ID has been correlated with a track

2.2.7 *Subroutine STAB*1

Subroutine STAB1 is called by the MULSIM executive routine. Its primary purpose is to produce the deck-plane coordinate of all the targets/platforms in the deck-plane coordinate system of a platform (ISHIP) which is designated in the calling sequence. The subroutine is entered with the stabilized coordinates and the current roll and pitch of platform ISHIP. With this information, it is a simple trigonometric exercise to rotate the stabilized coordinates into the deck-plane system. Table 2.8 lists the variables used in STAB1 in the order of their appearance in the program listing.

Table 2.8 — Variables in STAB1

Fortran Variable	Description
NT	Number of targets in scenario
NS	Number of platforms in scenario
ISHIP	Platform under consideration
ROLL(J), PITCH(J), (SHD(J)	Current roll, pitch, and heading of platform J (deg)
AZ(I,J), EL(I,J)	Current azimuth and elevation of target I in platform J's stabilized coordinate system (rad)
XX(I,J), YY(I,J)	Direction cosines of target I's position
SS(I,J)	Vector in platform J's deck-plane coordinate system
AZD(I,J), ELD(I,J)	Azimuth and elevation of target/platform I in platform J's deck-plane coordinate system

2.2.8 *Subroutine NOISY*

Subroutine NOISY is called by the MULSIM executive routine to provide the model with the noisy stabilized coordinates of every detection in a designated radar sector. The first step in the process is to go through each range bin in the sector and select individual detections from the range bin under consideration.

If there are detections in a particular range bin, the next step is to identify the targets they correspond to through the IDTA array and load the XYZTRU array with the true rectangular stabilized coordinates for future reference. This is more or less preliminary to the primary function of NOISY. The nucleus of NOISY is the STAB2 subroutine. Therein, samples are selected from a normal noise distribution derived from a random number generator. The samples are then added to the deck-plane coordinates, and the noisy deck-plane coordinates (range, azimuth, elevation) are transformed to the stabilized coordinate system and returned to NOISY. NOISY next takes the noisy stabilized range, azimuth, and elevation and determines the rectangular coordinates of the detection. These are loaded in the XYZMS file. The TMS file, which contains the time at which the detection was made, is loaded with the sector-crossing time. This process is repeated until all of the detections in sector ISEC have been considered.

Table 2.9 describes the variables used in NOISY listed in the order of their appearance in the listing. Figure 2.6 is a flowchart of the subroutine.

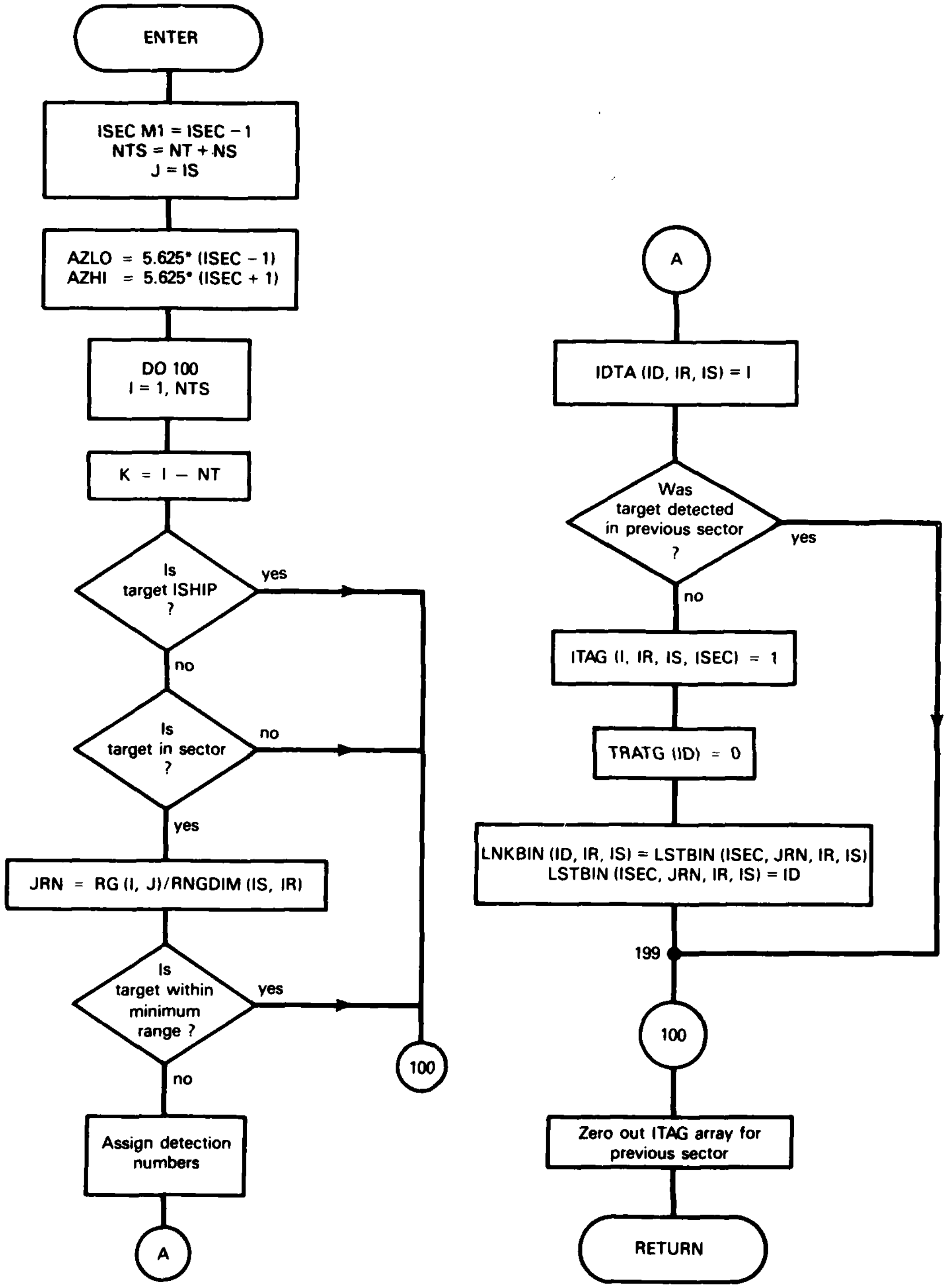

Fig. 2.5 — Subroutine DETFIL

Table 2.9 — Variables in NOISY

Fortran Variable	Description
IR, IS, ISEC	Radar, platform, and sector to be considered
JRN	Range bin number
LSTBIN(I,J,K,L)	Array containing identification number of last detection made in range bin J of sector I by radar K on platform L
IDTA(I,J,K)	Array containing target number of target that corresponds to detection I, made by radar J on platform K
XYZTRU(I,J,K,L)	True rectangular coordinates of detection I, made by radar K on platform L J = 1 x-coordinate = 2 y-coordinate = 3 z-coordinate
RG(I,J), AZ(IJ), EL(IJ)	Range, azimuth, elevation of target I with respect to platform J (m and rad). Values are true stabilized before call to STAB1; noisy afterwards.
XYZMS(I,J,KL)	Noisy stabilized rectangular coordinates of detection I made by radar K on platform L J = 1 x-coordinate = 2 y-coordinate = 3 z-coordinate
TMS(I,J,K)	Time at which detection I was detected by radar J on platform K
TMRK(I,J,K)	Time at which radar J on platform K passes from sector I to (I + 1).
LNKBIN(I,J,K)	Linking device containing number of the detection that was made prior to detection I and is in the same range bin as detection I. J and K represent the radar and platform number respectively.

2.2.9 *Subroutine STAB2*

Subroutine STAB2 is called by subroutine NOISY for a designated target, ship, and radar. Its primary function is to inject noise into the true deck-plane range, azimuth, and elevation and to transform the noisy measurements to the stabilized coordinate system. This is accomplished by selecting samples from a normal noise distribution derived from a random number generator (VRANF), weighting the samples by the standard deviation, and adding the result to the true deck-plane range, azimuth, and elevation. The transformation to the stabilized coordinate system is a simple 3-axis rotation involving the ship's roll, pitch, and heading.

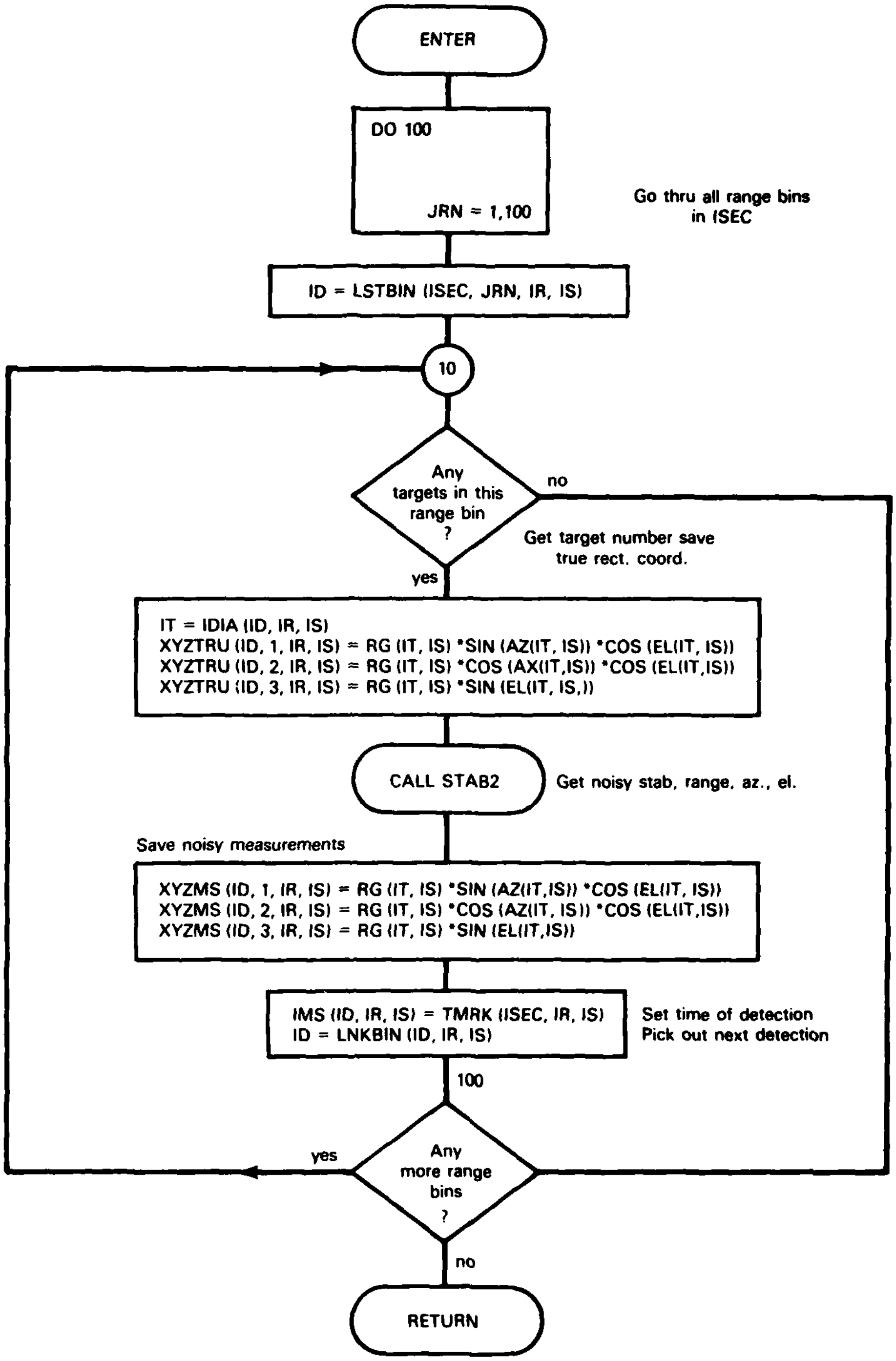

Fig. 2.6 — Subroutine NOISY

Table 2.10 lists the variables in the order that they are presented in the listing.

Table 2.10 — Variables in STAB2

Fortran Variable	Description
I,J,K	Designated target, platform, and radar
AZR, ELR	Noisy deck-plane measurements (rad)
AZND(I,J,K) ELND(I,J,K) RNND (I,J,K)	Noisy deck-plane azimuth, elevation, and range measurements for target I as measured by radar K on platform J
ROLL(J), PITCH(J)	Current roll and pitch of platform J
X,Y,Z	Direction cosines of target in stabilized system that has x-axis pointed in direction of ship's motion
SHD(J)	Current heading of platform J (deg)
AZ(I,J), EL(I,J) RG(I,J)	Stabilized azimuth, elevation, and range of target I with respect to platform J. True quantities are replaced by noisy quantities in these arrays before leaving STAB2.

2.3 Track Correlation/Integration System

2.3.1 *Subroutine PREDIC*

Subroutine PREDIC is called by the MULSIM executive routine. Its primary purpose is to keep the sector track files current; i.e., when a target changes sectors it must be deleted from the track file of the sector it has just left and added to the sector it has just entered.

Tracks are placed in track files according to the sector in which they are currently located. For example, after the detections from sector J (see Fig. 2.7) are placed in the appropriate detection files, the question arises as to which tracks are candidates for correlation with these detections. This question is not answered immediately. The correlation process takes place in sector (J-2) to account for delays in the system. However, before attempting the correlation process an intermediate bookkeeping step is required to keep the sector files current. The predicted positions of the tracks in the (J-1) sector file at the (J-1) sector crossing time are determined to see if they are still located in sector (J-1). Adjustments are made to the sector files to reflect any change in their position at this time.

Each ship maintains two sector track files, i.e., for each sector, each ship maintains what is referred to as a dummy track file,* which is used for local purposes, and a multiple platform track (MPT) file, which is used for communicating with other ships via the link.

*The dummy file is used for two reasons: (a) because there may be tentative tracks that have not gone out over the link, and (b) because provisions must be made for the system degrading to a single unit.

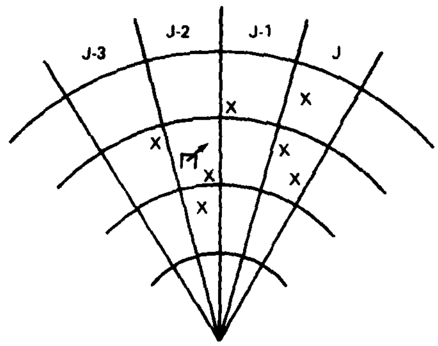

Fig. 2.7 — Locating tracks and detections

The first step taken by PREDIC in this process is to interrogate the dummy sector file (DUMSX) to determine if there are any tracks in sector ISEC. If there are no tracks, control is immediately returned to the executive routine. Otherwise the first track in the DUMSX file is examined to determine if it is a track that platform IS is responsible for updating. The X(I) file is next loaded with position and velocity coordinates from the XSMϕ(I,J,K) tracking file. For those tracks that come from other platforms the position coordinates and velocity components are run through the appropriate transformations by the TRANSF and VTRANS subroutines.

The next step is to determine the predicted position of the track at a time corresponding to the sector-crossing time (TMRK). The rectangular coordinates of the predicted position are stored in the XYZDUM array and the predicted range, azimuth, and elevation are stored in the RAEDUM array.

The bulk of the remaining logic in the subroutine is concerned with determining which sector contains the predicted track position. If the track (NT) remains in sector ISEC, no further action is taken. When the track has moved to a new sector, subroutine DUMDRP is called to remove it from sector ISEC's track file, and subroutine DUMNEW is called to place it in the track file of the new sector. This process is repeated until all the tracks in sector ISEC have been considered.

Table 2.11 contains the variables used in PREDIC in the order of their appearance in the program listing.

Table 2.11 — Variables in PREDIC

Fortran Variable	Description
ISEC, IR, IS	The sector, radar, and platform under consideration
DUMSX(I,J)	Array containing the identification number of the last track to be placed in sector I dummy track file on platform J
TMRK(I,J,K)	Sector-crossing time; i.e., the time at which radar J on platform K passes from sector I to sector I + 1.
TRKST(NT,IS)	File linking dummy tracks and MPT tracks; contains the MPT number of track NT from platform IS
PTFST(NT,IS)	File containing the platform number of the platform that is responsible for updating track NT from platform IS's dummy track file.
XSMO(I,MT,KS)	Smoothed position and velocity coordinates of track MT that reside in platform KS's MPT file I = 1, *x*-coordinate; I = 2, *x*-coordinate = 3, *y*-coordinate; = 4, *y*-coordinate = 5, *z*-coordinate; = 6, *z*-coordinate
XYZDUM(NT,I,IS)	Predicted rectangular position coordinates of track NT from platform IS's dummy file I = 1, *x*-coordinate = 2, *y*-coordinate = 3, *z*-coordinate
RAEDUM(NT,I,IS)	Predicted range, azimuth, and elevation of track NT with respect to platform IS's stabilized coordinate system. I = 1, range = 2, azimuth = 3, elevation

2.3.2 *Subroutine CORRAS*

2.3.2.1 Introduction

Subroutine CORRAS is called by the MULSIM executive routine. The primary purpose of the CORRAS subroutine is to identify those detections from a designated radar/platform that correlate with the tracks in a designated sector and to resolve conflicts that might arise when more than one track is correlated with the same detection. The latter function is referred to as the association process. A macro flowchart outlining the procedure adopted to correlate and associate detections with a group of tracks from a given azimuth sector is shown in Fig. 2.8.

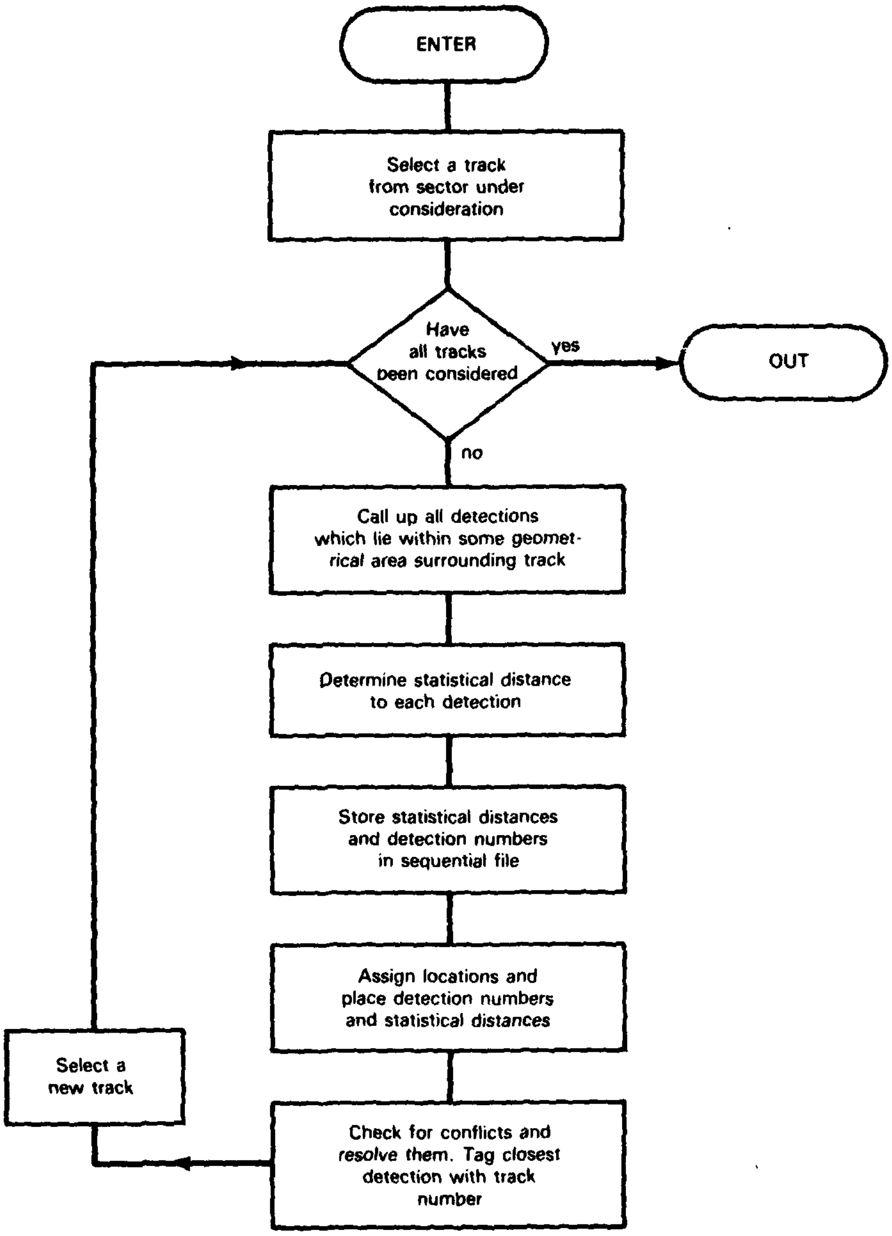

Fig. 2.8 — The correlation and association process

Basically what is involved in the correlation process is the selection of detections in some geometrical area surrounding a predicted track location. These detections are then ordered according to a previously defined statistical distance to the track under consideration and are stored in a retrievable file for use in the association process.

The association process is concerned with resolving conflicts that occur during correlation. Conflicts occur when two or more tracks have the same detection at the top of their detection files. The association process resolves these conflicts by comparing statistical distances.

2.3.2.2 The Correlation Process

The actual correlation process is preceded by the selection and location of tracks. This is detailed in Fig. 2.9. The track number (NT) at the top of the dummy sector file is selected to begin the process. The range bin (JRNG) that contains the predicted position of the track is then determined, and this is used to correlate the track with detections in its range-azimuth bin and in all adjacent range-azimuth bins; i.e., nine range-azimuth bins in all are considered. After the association process the next track in the sector dummy file is selected and the process is repeated until there are no tracks left to be considered in the sector clutter file (NT = 0).

The correlation process begins by considering the detections located in azimuth-range bin (I-1, J-1). See Fig. 2.10. The detection numbers of the detections from each range-azimuth bin have been placed in linked files. The first detection number (IDETNO) for bin I,J is obtained from LSTBIN (I,J,IRAD,ISHIP). When a value of zero is obtained from this file, it is an indication that there are no further detections to be considered in bin I,J. As each detection is drawn from the LNKBIN file, the statistical distance between the detection and the track under consideration is determined by subroutine COVOWN. This is used to order the detections in a sequential file IRADET(J), with the detection number having the smallest statistical distance placed in IRADET(1). This process is repeated for all adjacent bins and for the bin that contains NT. The net result is two sequential files: IRADET(J) containing the ordered detection numbers and SDIST(J) containing the ordered statistical distances. The number of detections (K) correlating with NT is also recorded. The sequential files are not tagged with an identifier relating them to NT because these values will be placed in a linked file during the association process, thereby allowing greater flexibility in retrieval and storage. The correlation process is detailed in Fig. 2.9.

2.3.2.3 The Association Process

The association process begins with placing the statistical distances and detections, which have been correlated with the track under consideration, in linked files. The detection numbers are placed in ISTOR(LOC) and the statistical distances in SD(LOC). The detections are located in these files with a pointer DETSX(LOC). This gives the location (LOC) of the correlated detection that has the smallest statistical distance to the track under consideration. The location of the detection with the smallest statistical distance is found from the link DETID(LOC), i.e., DETID(LOC) contains the location of the detection that follows the one found in LOC. As detection numbers and statistical distances are fed into the ISTOR

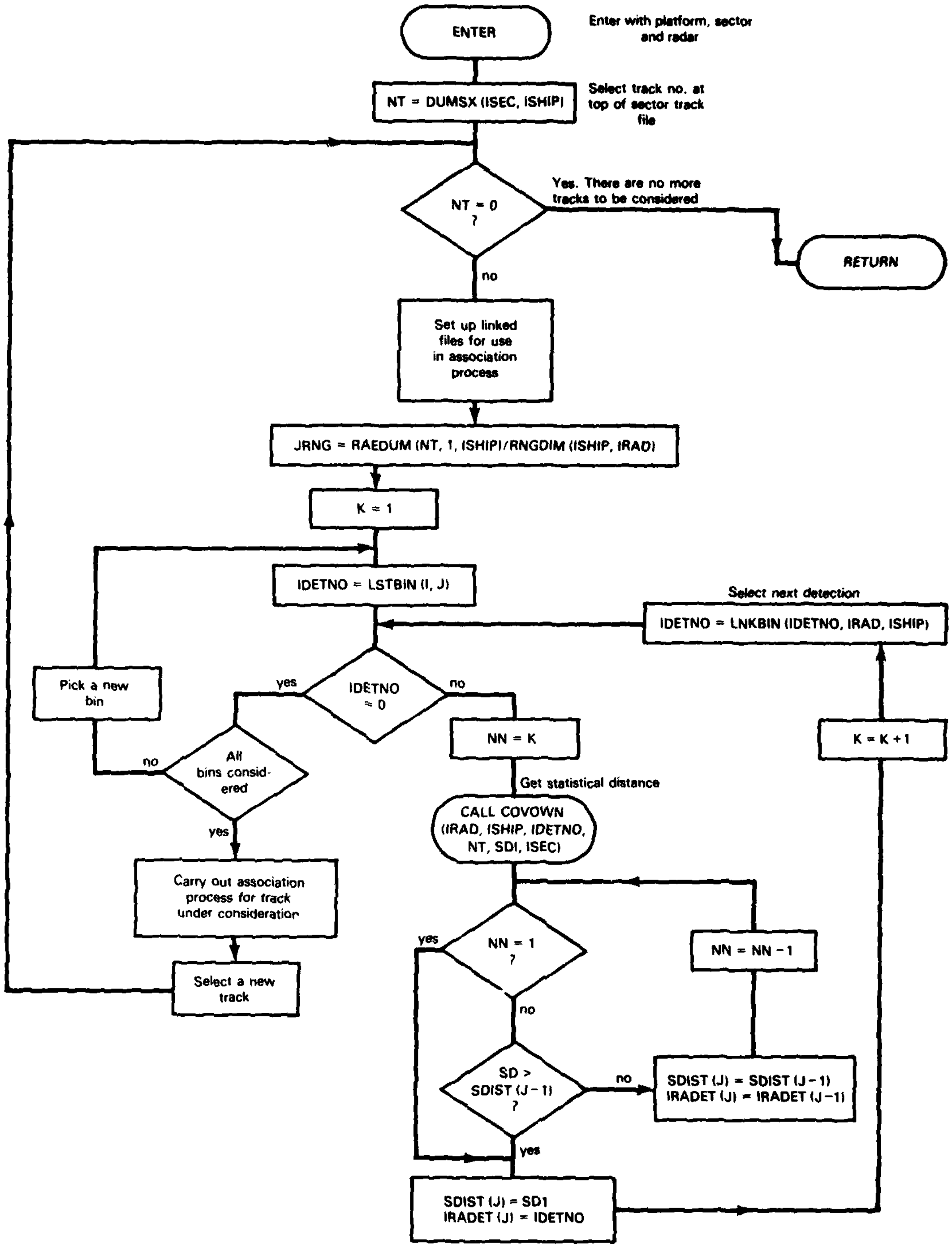

Fig. 2.9 — The correlation process

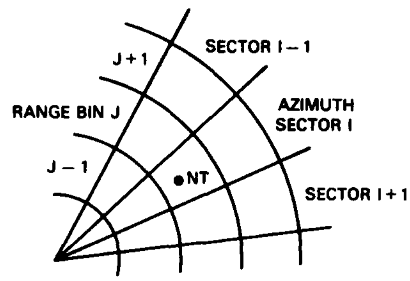

Fig. 2.10 — Azimuth—range bins

and SD files, locations in these files are provided by the NEWLOC subroutine. Flowcharts of the NEWLOC subroutine and the association process are found in Figs. 2.11 and 2.12 respectively.

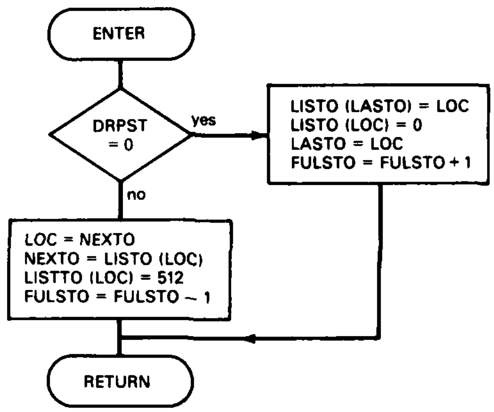

Fig. 2.11 — Subroutine NEWLOC

After the correlated detections and statistical distances have been placed in the linked files, it is possible to proceed with the resolution of conflicts. This begins with the selection of the first detection number (IDETNO) from the linked file. For each detection there is a one-to-one relationship with some track. This is established through the use of the TRATG file. If TRATG(IDETNO) is zero, it means that a relationship has not previously been established between IDETNO and some track. In this case TRATG(IDETNO) is assigned the value NTA. If TRATG(IDETNO) has been previously assigned another track number (NTT), this means that a conflict exists. The conflict is resolved by comparing statistical distances. If the statistical distance between NTA and IDETNO is less than that between NTT and IDETNO, then TRATG(IDETNO) is assigned the value NTA, and the next detection in the linked file of detections that correlated with NTT is examined for a possible conflict with

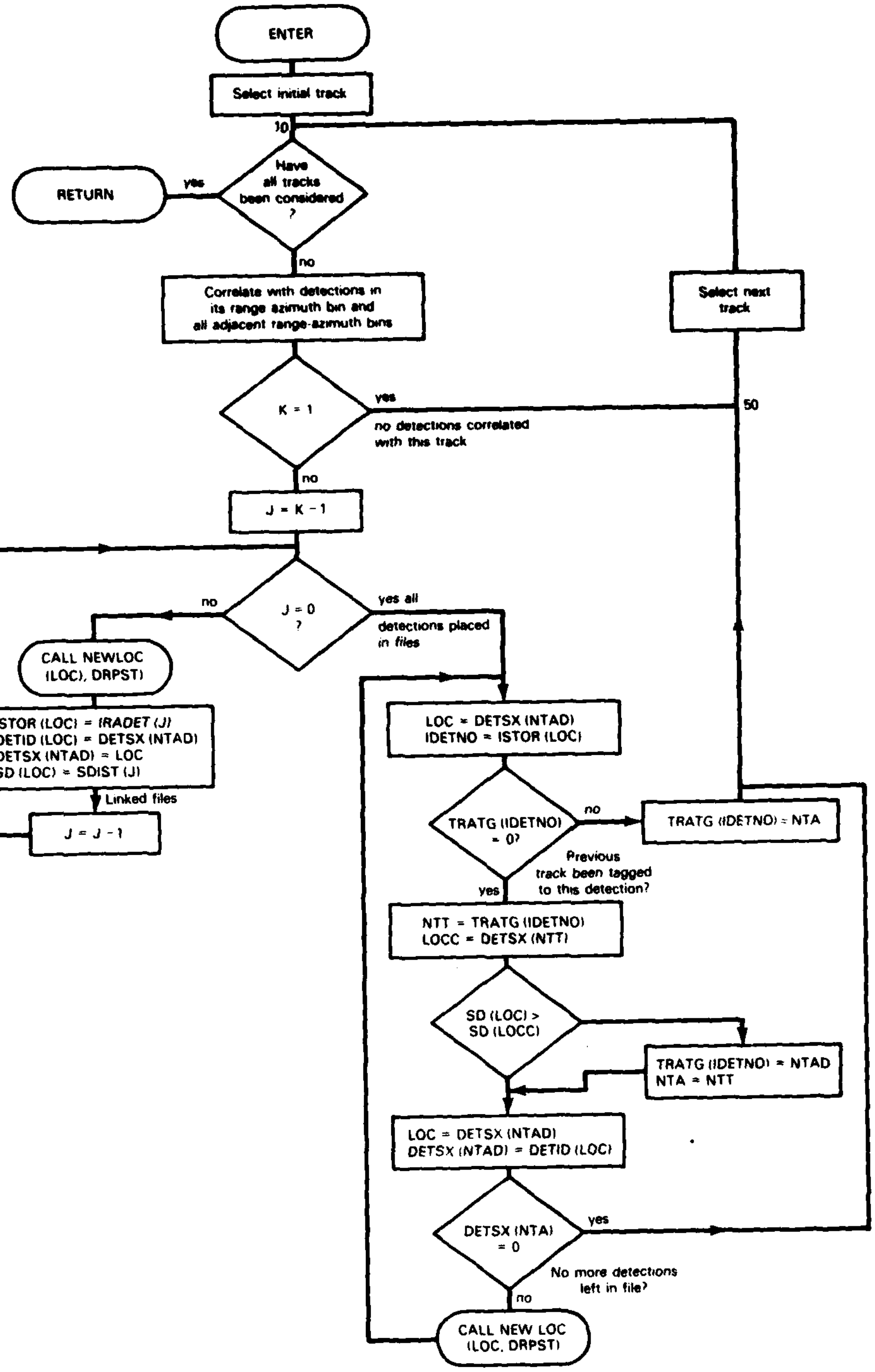

Fig. 2.12 — The association process

other tracks. This process will continue until a one-to-one relationship is established between a track and a detection or there are no more detections left in a linked detection file. If the statistical distance between NTA and IDETNO is not less than that between NTT and IDETNO, the process reverts to the next detection in the linked file associated with NTA and again an attempt is made to establish a one-to-one relationship. As each conflict is resolved, the location of the detection which correlated with the track having the larger statistical distance is made available for future storage in the ISTOR and the SD file. This is accomplished by calling the NEWLOC subroutine with IDRP set equal to zero.

When all tracks in the designated sector have been correlated with the given detections and all conflicts have been resolved, the process is complete.

The Fortran variables used in CORRAS are listed in Table 2.12.

Table 2.12 — Variables in CORRAS

Fortran Variable	Description
ISHIP, ISEC, IRAD	Platform, sector, and radar under consideration
DUMSX(I,J)	Array containing the trade number of last track to be placed in platform J's Ith sector file.
DETSX(NT)	Array that identifies the locations in the ISTOR and SD files that contain the ID number and the statistical distance to tract NT of the detection with the smallest statistical distance to NT
DETID(LOC)	Linking device that gives the location of detection and statistical distance that is next smaller than the one found in LOC
NT	Track number
JRNG	Range-bin identifying number
RAEDUM(NT,I,K)	Predicted range of track NT from platform K's dummy track file
RNGDIM(ISHIP, IRAD)	Range dimension assigned to range bins associated with radar IRAD on platform ISHIP
LSTBIN(I,J,K,M)	Array containing identification number of last detection made in range bin J of sector I by radar K on platform M
LNKBIN(I,J,K)	Linking device containing the number of the detection that was made prior to detection I and is in the same range bin as detection I. J and K represent the radar and platform number, respectively

Table 2.12 (Concluded) — Variables in CORRAS

Fortran Variable	Description
SDIST(NN)	Array containing statistical distances from track under consideration to all the detections that are correlated with it. SDIST(1) is smallest.
IRADET(NN)	Array containing detection numbers of detections that are correlated with track under consideration. IRADET(J) corresponds with SDIST(J).
DUMID(NT,ISHIP)	Linking device containing the track number of the track that was placed in sector track file prior to track NT
ISTOR(LOC)	Linked file containing detection numbers of correlated detections. Similar to IRADET temporary file. Accessed by DETSX pointer and DETID linking device.
SD(LOC)	Linked file containing statistical distances between detection in ISTOR and track designated by DETSX.
TRATG(ID)	Flag indicating status of detection ID TRATG(ID) = 1; detection has been correlated with some track. TRATG(ID) = 0; detection has not been correlated with any track.
IDRP	Flag for subroutine NEWLOC IDRP = 1; find a new location IDRP = 0; vacate location

2.3.3 *Subroutine COVOWN*

Subroutine COVOWN is called by subroutine CORRAS. Its primary purpose is to provide the KALMAN subroutine with the measurement covariance matrix in the stabilized coordinate system of the tracking platform. If the tracking platform is not the detecting platform, the calculations are made by subroutine COVLNK, which is discussed in section 2.3.4. The measurement covariance is used by KALMAN to determine the statistical distance from the detection to the target.

The subroutine is presented with the standard deviations of the range, azimuth, and elevation. It is assumed that there is no cross-correlation between the three measurements, hence, the covariance matrix as represented by the standard deviations is diagonalized. Since the measurements are made in the deck-plane coordinates, it is COVOWN's task to transform this matrix to the stabilized coordinate system of the tracking platform. Because the deck-plane coordinate system is continually in rotational motion with respect to the stabilized coordinate system, this procedure must be carried out each time a statistical distance is requested by CORRAS.

The process can be represented mathematically as follows: the position vector in the deck-plane coordinates can be expressed as a truncated Taylor's series

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \overline{x} \\ \overline{y} \\ \overline{z} \end{bmatrix} + \begin{bmatrix} \partial x/\partial R & \partial x/\partial \eta & \partial x/\partial \alpha \\ \partial y/\partial R & \partial y/\partial \eta & \partial y/\partial \alpha \\ \partial z/\partial R & \partial z/\partial \eta & \partial z/\partial \alpha \end{bmatrix} \begin{bmatrix} N_r \\ N_\eta \\ N_\alpha \end{bmatrix}, \tag{2.3.1}$$

where

$(\overline{x},\overline{y},\overline{z})$	is the mean or true position of the target in the platform's deck-plane coordinate system
(x, y, z)	is the target position vector in the platform's deck-plane coordinate system, based on measurements made at the platform
$(\partial x/\partial R, \partial x/\partial \eta$, etc.)	are the partial derivatives of the components of the position vector with respect to changes in the measurements made at the platform
(N_r, N_η, N_α)	represent noise in the measurements made by the platform's radar
(R, η, α)	represent the range, elevation, and azimuth measurements made by the platform.

If vector matrix notation is used, Eq. (2.3.1) may be rewritten as

$$\vec{X} = \vec{\overline{X}} + \widetilde{A}\,\vec{N}. \tag{2.3.2}$$

The deck plane coordinates can be transformed from the stabilized coordinates by the $H(I,J)$ and $P(I,J)$ matrices. The H matrix accounts for the platform's heading and the P matrix accounts for the platform's pitch and roll. This can be expressed as

$$\widetilde{H}\cdot\widetilde{P}\cdot\vec{X} = \widetilde{H}\cdot\widetilde{P}(\vec{\overline{X}} + \widetilde{A}\cdot\vec{N}) \tag{2.3.3}$$

or

$$\widetilde{H}\cdot\widetilde{P}(\vec{X} - \vec{\overline{X}}) = \widetilde{H}\cdot\widetilde{P}\cdot\widetilde{A}\cdot\vec{N}. \tag{2.3.4}$$

The left side of Eq. (2.3.4) represents the measurement error in the stabilized coordinate system and it is the covariance of this quantity that is required; i.e.,

$$\mathrm{cov}(M) = E(\widetilde{H}\cdot\widetilde{P}\cdot\widetilde{A}\cdot\vec{N}\cdot\vec{N}^T\cdot\widetilde{A}^T\cdot\widetilde{P}^T\cdot\widetilde{H}^T)$$

or

$$= \widetilde{H}\cdot\widetilde{P}\cdot\widetilde{A}\cdot\vec{N}\cdot\vec{N}^T\cdot A^T\cdot P^T\cdot\widetilde{H}^T. \tag{2.3.5}$$

This is represented by the $COVM(I,J)$ matrix in the listing; the product $\widetilde{H} \cdot \widetilde{P} \cdot \widetilde{A}$ is given by the $(AJS(I,J)$ matrix, and $\vec{N} \cdot \vec{N}^T$ is the diagonalized covariance represented by the standard deviations in range, azimuth, and elevation measurements.

The covariance matrix, the time at which the detection was made, the track with which is has been correlated, and the measured stabilized coordinates of the detection are presented to the KALMAN subroutine which determines the statistical distance. This completes the process and control is returned to CORRAS. A flowchart of the process and a list of Fortran variables are in Fig. 2.13 and Table 2.13, respectively.

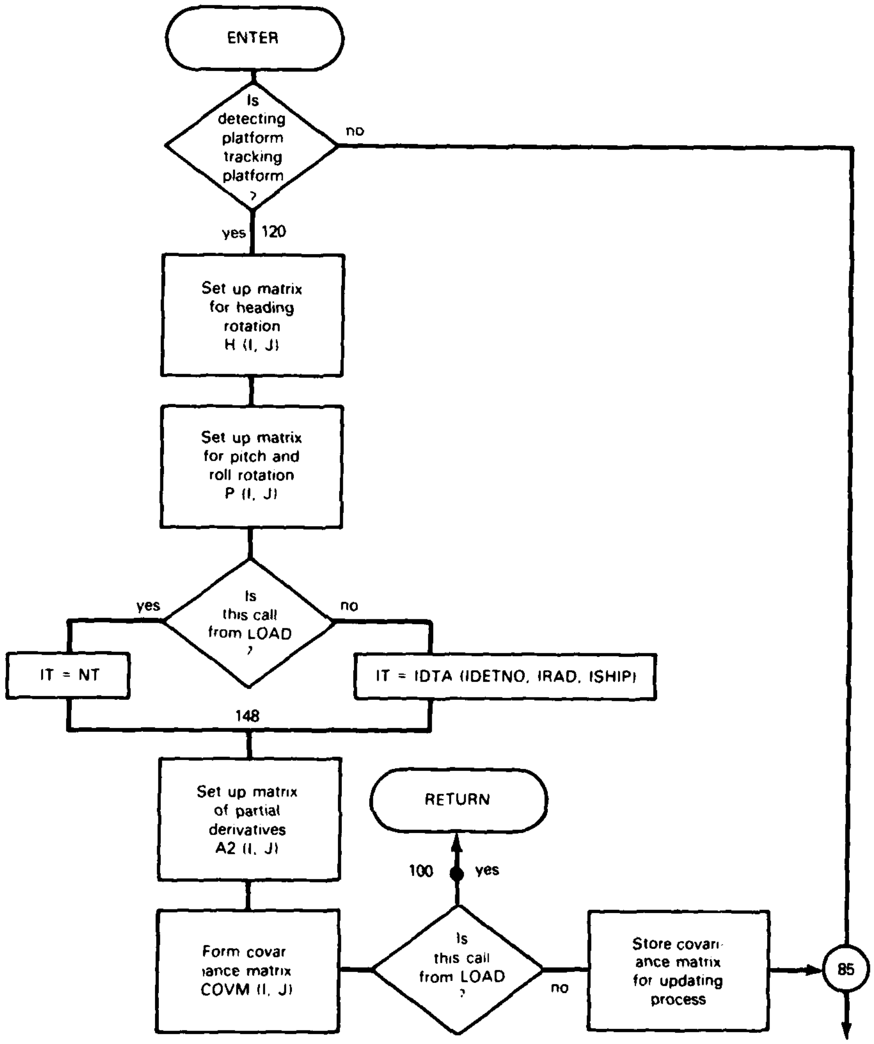

Fig. 2.13 — Subroutine COVOWN

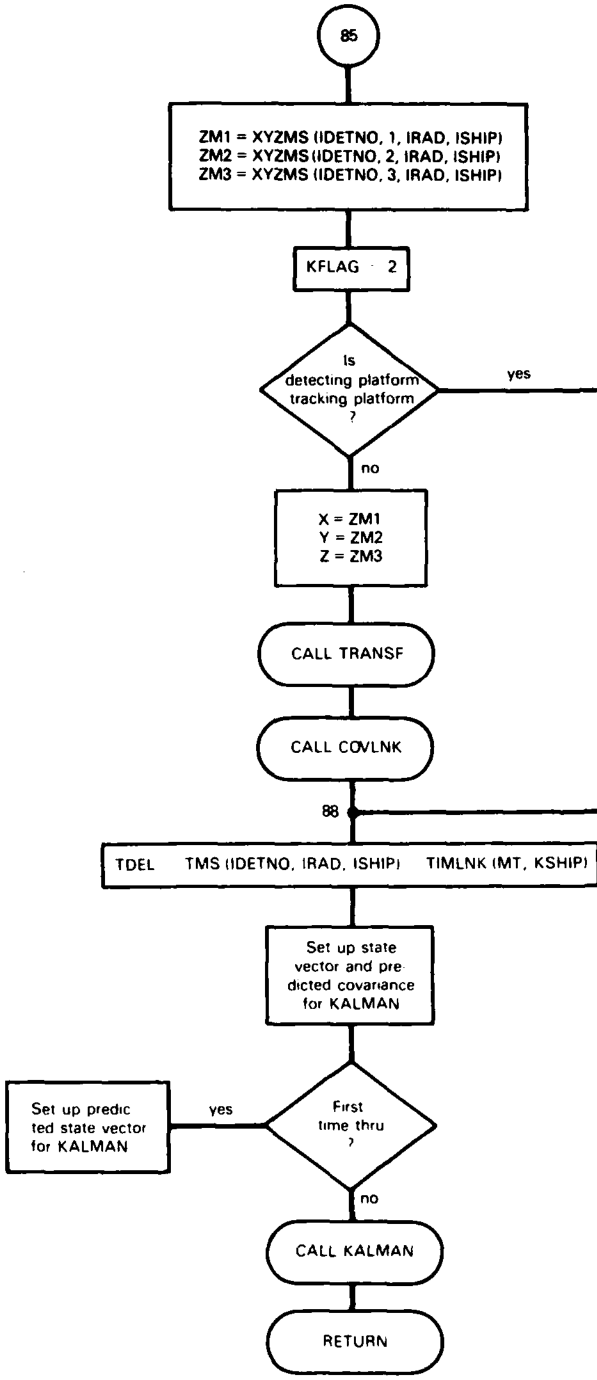

Fig. 2.13 (Concluded) — Subroutine COVOWN

Table 2.13 — Variables in COVOWN

Fortran Variable	Description
IRAD	Number of radar making detection
ISHIP	Number of platform making detection
IDETNO	Number of detection under consideration
NT	Number of track that correlates with IDETNO
SD	Statistical distance
ISEC	Sector in which detection was made
MT	Multiple platform track number corresponding to NT
TRKST(I,J)	File that contains MPT number of dummy track I from platform J
PTFST(I,J)	File that contains platform number of the platform that is tracking dummy track I from platform J
SHD(I)	Current heading of ship I
H(I,J)	Array that accounts for platform's heading in coordinate transformation
P(I,J)	Array that accounts for platform's roll and pitch in coordinate transformation
ROLL(J), PITCH(J)	Current roll and pitch of platform J
ELND(IT,ISHIP,IRAD) AZND(IT,ISHIP,IRAD) RNND(IT,ISHIP,IRAD)	Noisy deck-plane measurements of target IT as made by IRAD on platform ISHIP.
A2(I,J)	Array of partial derivatives of Ith component of position vector with respect to Jth component of measurement
COVM(I,J)	Measurement covariance matrix
COVMS(N,I,J,K,L)	Measurement covariance matrix associated with detection N, radar K, and platform L
XYZMS(I,J,K,L)	Measurement position vector of detection I, radar K, platform L
KFLAG	Indicator that tells KALMAN that call is for statistical distance calculation (KFLAG = 2) or updating (KGLAG = 1)
TMS(I,J,K)	Time at which detection I was made by radar J on platform K
TIMLNK(I,J)	Time at which track I was last updated by platform J
XSMO(I,J,K)	Smoothed position vector of track J as determined by platform K

2.3.4 *Subroutine COVLNK*

Subroutine COVLNK is called by subroutine COVOWN and subroutine UPDATE. When the tracking platform is not the detecting platform, COVLNK is called to produce the measurement covariance matrix in the tracking platform's stabilized coordinate system. The measurement covariance is used by KALMAN to determine the statistical distance from the detection to the target.

As with COVOWN the problem is to transform the measurement covariance matrix to the stabilized coordinate system of the tracking platform. The process is further complicated by uncertainties in the relative location of the two platforms. Mathematically it can be represented as follows: the position vector with respect to the supposed location of the tracking platform's origin is given by

$$\vec{X}_2 + \widetilde{T}\,[\vec{S}_1 + \Delta\vec{X}_1] + \widetilde{U} \tag{2.3.6}$$

where $\widetilde{T}$ and $\widetilde{U}$ are rotational and translational matrices, respectively. $\vec{S}_1$ is the location of the target in the detecting platform's stabilized coordinate system and $\Delta\vec{X}_1$ represents the inexact location of the detecting platform. The position vector of the target in the tracking platform's stabilized coordinate system is given by

$$\vec{S} = \vec{X}_2 - \vec{\Delta X}_2 \tag{2.3.7}$$

$$= \widetilde{T}\,[\vec{S}_1 + \vec{\Delta X}_1] + \widetilde{U} - \vec{\Delta X}_2 \tag{2.3.8}$$

$$= \widetilde{T}\,[\widetilde{H}\cdot\widetilde{P}\cdot\vec{D}_1 + \vec{\Delta Y} + \Delta Y_1] + \widetilde{U} - \vec{\Delta X}_2 \tag{2.3.9}$$

where $\widetilde{H}$ and $\widetilde{P}$ are defined as in COVOWN and $\vec{D}_1$ is the position vector of the target in the detecting platform's deck-plane system. The stabilized position vector $(\vec{S}_2)$ can also be expressed as a truncated Taylor's series

$$\begin{bmatrix} s_2 \\ t_2 \\ u_2 \end{bmatrix} = \begin{bmatrix} s_2 \\ t_2 \\ u_2 \end{bmatrix} + \begin{bmatrix} \partial s_2/\partial\Delta x_1 & \partial\Delta_2/\partial\Delta y_1 & \partial s_2/\partial\Delta z_1 \ldots \partial s_2/\partial\Delta x_2 & \partial s_2/\partial\Delta y_2 & \partial s_2/\partial\Delta z_2 \\ \partial s_2/\partial\Delta x_1 & \partial s_2/\partial\Delta y_1 & \partial s_2/\partial\Delta z_1 \ldots \partial s_2/\partial\Delta x_2 & \partial s_2/\partial\Delta y_2 & \partial s_2/\partial\Delta z_2 \\ \partial s_2/\partial\Delta x_1 & \partial s_2/\partial\Delta y_1 & \partial s_2/\partial\Delta z_1 \ldots \partial s_2/\partial\Delta x_2 & \partial s_2/\partial\Delta y_2 & \partial s_2/\partial\Delta z_2 \end{bmatrix} \begin{bmatrix} N_{\Delta x_1} \\ N_{\Delta y_1} \\ N_{\Delta z_1} \\ N_{r_1} \\ N_{\eta_1} \\ N_{\alpha_1} \\ N_{\Delta x_2} \\ N_{\Delta y_2} \\ N_{\Delta z_2} \end{bmatrix} \tag{2.3.10}$$

or

$$\vec{S}_2 = \bar{\vec{S}}_2 + \widetilde{A} \cdot \vec{N} \tag{2.3.11}$$

and

$$\vec{S}_2 - \bar{\vec{S}}_2 = \widetilde{A} \cdot \vec{N} \tag{2.3.12}$$

where the elements of $\widetilde{A}$ are found from differentiating Eq. (2.3.9). The left side of Eq. (2.3.12) represents the error in the position of the target in the tracking platform's stabilized coordinate system. Errors are induced by inaccurate measurements from the detecting platform and the inexact locations of both platforms. It is the covariance of this quantity that is needed for determining the statistical distance and updating, i.e.,

$$\mathrm{cov}\,(\overline{M}) \;=\; E\,(\widetilde{A}\cdot\vec{N}\cdot\vec{N}^T\widetilde{A}^T) \tag{2.3.13}$$

or

$$= \;\widetilde{A}\cdot\overline{\vec{N}\cdot\vec{N}^T}\cdot\widetilde{A}^T \tag{2.3.14}$$

which is represented by the COV$M(I,J)$ matrix in the program listing and $\overline{\vec{N}\vec{N}^T}$ is the diagonalized covariance matrix represented by the standard deviations of measurements and platform locations. The covariance matrix is returned to COVOWN or to UPDATE for the KALMAN subroutine. The Fortran and variables not used in COVOWN are listed in Table 2.14.

Table 2.14 — Variables Not Used in COVOWN

Fortran Variable	Description
KS	Detecting platform
ID	Detection number
IR	Radar number of radar making detection
JSHIP	Tracking platform
SLAT(I), SLOG(I)	Current latitude and longitude of platform I
T(I,J)	Transformation matrix for transforming from one platform's stabilized coordinate system to another
A(I,J)	Array of partial derivatives as defined in Eq. (2.3.10)
N(I)	Standard deviations of noise in measurement and platform positions

2.3.5 *Subroutine KALMAN*

Subroutine KALMAN is called by subroutines COVOWN to provide statistical distance information and by subroutine UPDATE to update the estimate of the state vector of a given track. The smoothing and estimation process is performed by a Kalman filtering algorithm similar to the 2D algorithm described in Ref. 3. The state equation in the tracking platform's stabilized coordinate system is given by

$$X(t+1) = \phi(t)\,X(t) + \Gamma(t)\,A(t) \tag{2.3.15}$$

where

$$X(t) = \begin{bmatrix} x(t) \\ x(t) \\ y(t) \\ y(t) \\ z(t) \\ z(t) \end{bmatrix}, \qquad \phi(t) = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\Gamma(t) = \begin{bmatrix} \tfrac{1}{2}T^2 & 0 & 0 \\ T & 0 & 0 \\ 0 & \tfrac{1}{2}T^2 & 0 \\ 0 & T & 0 \\ 0 & 0 & \tfrac{1}{2}T^2 \\ 0 & 0 & T \end{bmatrix} \text{ and } A(t) = \begin{bmatrix} a_x(t) \\ a_y(t) \\ a_z(t) \end{bmatrix} \qquad (2.3.16)$$

with $X(t)$ being the state vector at time t consisting of position and velocity components $x(t)$, $\dot{x}(t)$, $y(t)$, $\dot{y}(t)$, $z(t)$, and $\dot{z}(t)$, $t + 1$ being the next observation time, T being the time between measurements, and $a_x(t)$, $a_y(t)$, and $a_z(t)$ being random accelerations whose covariance matrix is $Q(t)$. The observation equation is

$$Y(t) = M(t)X(t) + N(t),$$

where

$$Y(t) = \begin{bmatrix} x_m(t) \\ y_m(t) \\ z_m(t) \end{bmatrix}, \qquad M(t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } N(t) = \begin{bmatrix} v_x(t) \\ v_y(t) \\ v_z(t) \end{bmatrix} \qquad (2.3.17)$$

with $Y(t)$ being the measurement at time t consisting of positions $x_m(t)$, $z_m(t)$, and $y_m(t)$; and $N(t)$ being zero mean noise whose covariance matrix is $R(t)$, provided by COVOWN or COVLNK.

The problem is solved recursively by first assuming that the problem is solved at time $(t-1)$. Specifically it is assumed that the best estimate $\hat{X}(t-1 \mid t-1)$ at time $(t-1)$ and its error covariance matrix $P(t-1 \mid t-1)$ are known, where the circumflex signifies an estimate and $\hat{X}(t \mid s)$ signifies that $X(t)$ is being estimated with observations up to $Y(s)$. The six steps involved in the recursive algorithm are as follows:

Step 1. Calculate one-step prediction

$$\hat{X}(t|t-1) = \phi(t-1)\hat{X}(t-1|t-1) \tag{2.3.18}$$

Step 2. Calculate the covariance matrix for one-step prediction,

$$P(t|t-1) = \phi(t-1)P(t-1|t-1)\tilde{\phi}(t-1) + \Gamma(t-1)Q(t-1)\tilde{\Gamma}(t-1) \tag{2.3.19}$$

Step 3. Calculate the prediction observation

$$\hat{Y}(t|t-1) = M(t)\hat{X}(t|t-1) \tag{2.3.20}$$

Step 4. Calculate the filter gain

$$\Delta(t) = P(t|t-1)\widetilde{M}(t)\ [M(t)P(t|t-1)\widetilde{M}(t) + R(t)]^{-1} \tag{2.3.21}$$

Step 5. Calculate a new smoothed estimate

$$\hat{X}(t|t) = \hat{X}(t|t-1) + \Delta(t)\ [Y(t) - \hat{Y}(t|t-1)] \tag{2.3.22}$$

Step 6. Calculate a new covariance matrix

$$P(t|t) = [I - \Delta(t)M(t)]P(t|t-1). \tag{2.3.23}$$

In summary, with an estimate $\hat{X}(t-1 \mid t-1)$ and its covariance matrix $P(t-1 \mid t-1)$, after a new observation $Y(t)$ is received and the six quantities in the recursive algorithm are calculated, a new estimate $\hat{X}(t \mid t)$ and its covariance matrix $P(t \mid t)$ are obtained.

For each track, the first two passes through KALMAN are used to initialize the filter. On the first pass the smoothed covariance is approximated by setting the appropriate elements equal to the measurement covariance matrix. On the second pass the definition of the smoothed covariance matrix, the current measurement covariance, and previous smoothed covariance and the time step are used to determine the elements of the new smoothed covariance matrix. When KALMAN is called for statistical distance before the first two passes are complete, the statistical distance is arbitrarily assigned a value of 10. The statistical distance as calculated by the model is the squared Mahalanobis distance [2] from the smoothed track position to the predicted track position plus the Mahalanobis distance from the smoothed position to the measured position. The Mahalanobis distance differs from the Euclidian distance in that a covariance matrix kernel is used instead of the identity matrix.

The Fortran variables equivalent to the algebraic variables used in Eqs. (2.3.18) through (2.3.23) are listed in Table 2.15.

Table 2.15 — Variables in Recursive Algorithm of Eqs. (2.3.18) to (2.3.23)

Fortran Variable	Description
XP(I)	Predicted state vector $\hat{X}(t \mid t-1)$
ZM(I)	Measurement vector $Y(t)$
PS(I,J)	Smoothed covariance matrix $P(t \mid t)$
COVM(I,J)	Measurement covariance matrix $R(t)$
TDEL	Time between measurements T
H(I,J)	Matrix $M(t)$ from observation Eq. (2.3.17)
WT(I,J)	Filter gain Δt
PHI(I,J)	State transition matrix $\phi(t-1)$
PP(I,J)	Predicted covariance matrix $P(t \mid t-1)$
COVS(I,J)	Covariance of random accelerations $Q(t)$
G(J,K)	State transition matrix $\Gamma(t-1)$
XS(I)	Smoothed state vector $\hat{X}(t \mid t)$

2.3.6 *Subroutine SORT*

Subroutine SORT is called by the MULSIM executive routine. Its primary purpose is to sort the detections associated with the tracks in a given sector into three distinct categories: those associated with tracks from participating platforms, those associated with tracks from the detecting platform's file, and those associated with tracks that another platform is responsible for updating.

SORT is presented with a designated platform, sector, and radar. The detections associated with each track in the sector file are processed according to which category the track falls into. If the track is a participating platform no further processing is required since platform updating is not dependent on radar detections; however, it is proposed in the future to use the detections of platforms to develop a bias removal scheme. If the track is assigned to another platform, the detecting platform's roll, pitch, and heading are placed in files for transmission over the link, and a random delay is added to the detection time to produce the time at which the detection information will be sent over the link. This information is next used by the contact selector subroutine (TIMCON) to select contacts for transmission over the link. Last, if the track is assigned to the detecting platform, the linked TESTO file is loaded with information for accessing the detection files. TESTO contains the identification number of all detections that have recently been associated with the track under consideration.

When all tracks in the sector track file have been considered, control is returned to the executive routine. A flowchart of the process and a list of Fortran variables are found in Fig. 2.14 and Table 2.16 respectively.

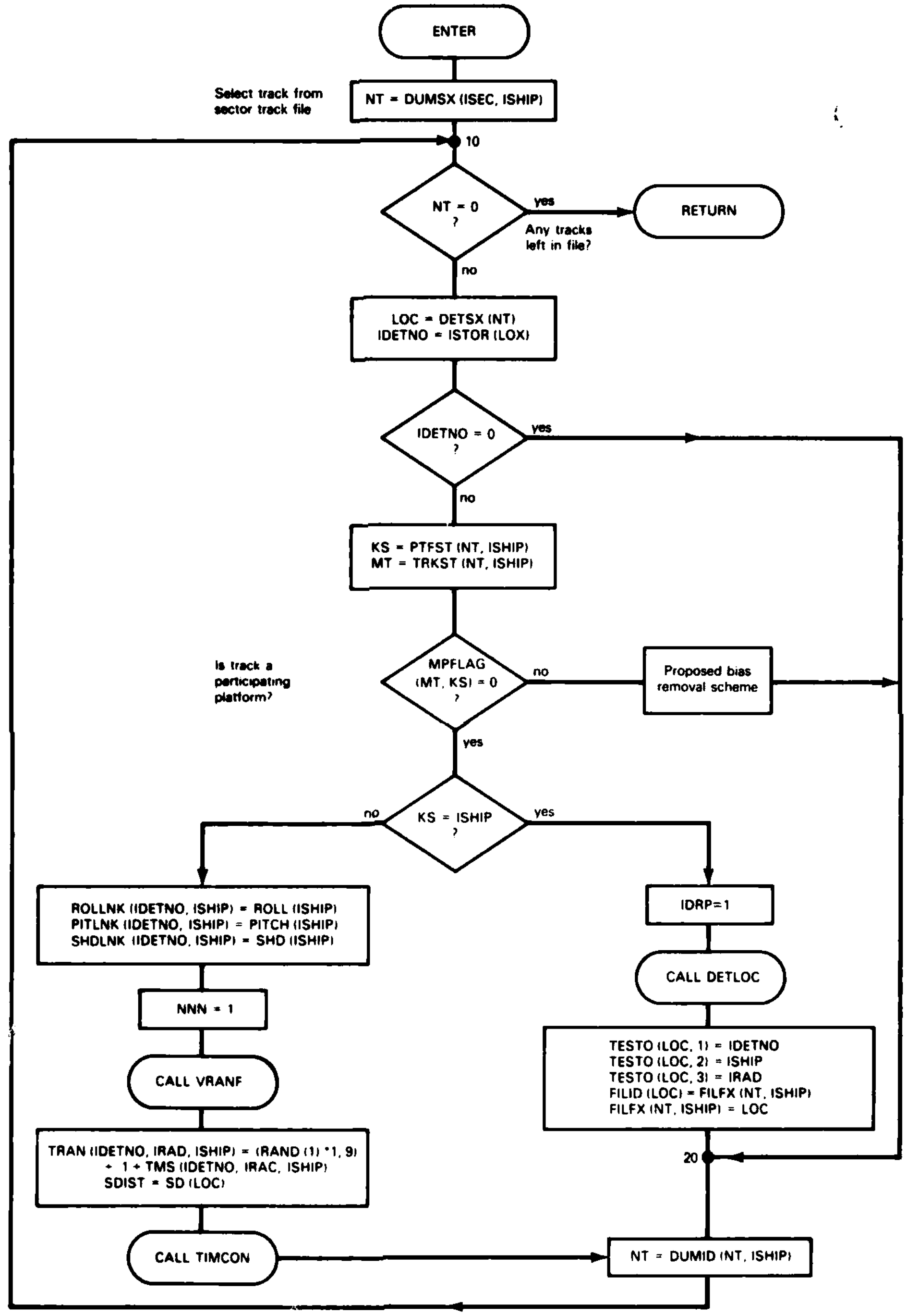

Fig. 2.14 — Subroutine SORT

Table 2.16 — Variables in SORT

Fortran Variable	Description
ISEC, ISHIP, IRAD	Sector, platform, and radar under consideration
DUMSX(ISEC,ISTOP)	Array that contains the track number of first track in platform ISHIP's sector track file
DETSX(NT)	Pointer that provides location of detection information on detection that has been associated with track NT
ISTOR(LOC)	File containing the number of the detection associated with track NT. LOC is provided by DETSX.
PTFST(NT,ISHIP)	Number of the platform responsible for updating track NT from ISHIP's dummy sector file
TRKST(NT,ISHIP)	Number of MPT track that corresponds to track NT from ISHIP's dummy sector file
ROLL(I) PITCH(I), SHD(I)	Current roll, pitch, and heading of platform I
TMS(I,J,K)	Time of detection for detection I in platform K's Jth sector
TESTO(LOC,I)	Linked file containing numbers of the detections associated with track NT. File also contains respective platform and radar numbers I = 1; detection number = 2; platform number = 3; radar number
FILFX(NT,I)	Pointer for accessing TESTO file. Gives location of last detection number, etc., to be associated with NT from platform I's dummy to be sector file
FILID(LOC)	Linking device that links locations of all of the detections associated with track NT

2.3.7 *Subroutine TIMCON*

Subroutine TIMCON is called by subroutine SORT. Its primary function is to act as a rudimentary contact selector; i.e., it decides which of the many detections should be selected for transmission over the communications link. This is accomplished by dividing time into 1-s intervals and defining five intervals as a time slot. When a target is detected, a flag (ISLOT) that identifies the associated track, the platform, and the time interval is changed from zero to one. When additional detections are considered, the time interval flags are examined. If a previous associated detection has been transmitted in the current time interval or in the four previous time intervals (which totals one time slot), then the detection is rejected. For those detections selected for transmission, the LNKSTO file is loaded with the detection number, the detecting platform's number, the detecting radar's number, the MPT number of the associated track, and the updating platform's number. The LNKSTO file is used in the updating process to identify the detections that are candidates for updating a specified track.

This first-come-first-served type of selector is obviously not the optimum contact selector. Other selectors are being considered and will be installed in the future. A flowchart of the process and a list of Fortran variables are found in Fig. 2.15 and Table 2.17 respectively.

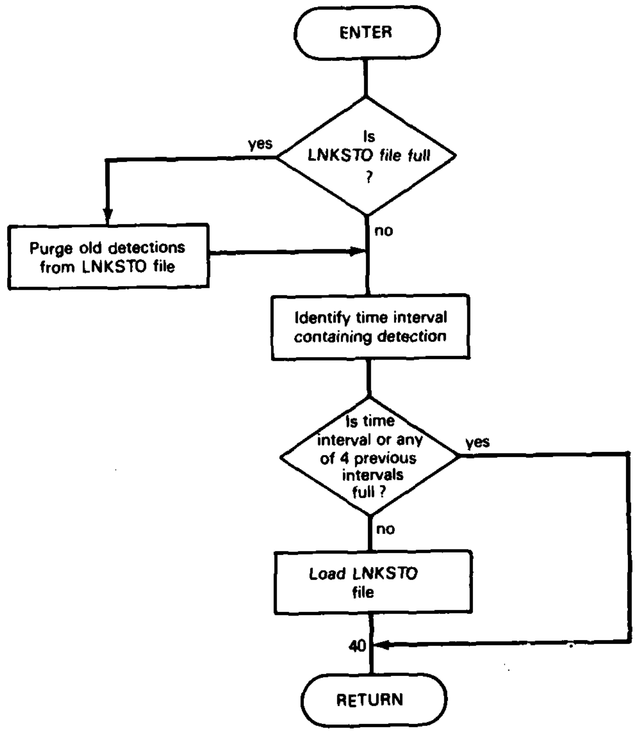

Fig. 2.15 — Subroutine TIMCON

Table 2.17 — Variables in TIMCON

Fortran Variable	Description
IDETNO	Number of the detection under consideration
MT	Number of the MPT track associated with IDETNO
KS	Platform responsible for updating track MT
IS,IR	Platform and radar making detection IDETNO
ISEC	Sector containing track MT
SDIST	Statistical distance between MT and IDETNO
FULLNK	Counter that keeps track of the number of detections in LNKSTO file
TMS(ID,IR,IS)	Detection time of detection ID, from radar IR on platform IS
TMRK(ISEC,IR,IS)	Sector-crossing time at sector ISEC
ITIME	Detection time truncated to an integer
MOD60	ITIME modulo 60 s
ISLOT(MT,KS,I)	Time slot I for track MT from platform KS
LNKFSX	Pointing device for accessing LNKSTO file
LNKID(LOC)	Linking device that links locations in LNKSTO
LNKSTO(LOC,I)	Linked file containing identification information on associated detections that are being sent over the link I = 1; detection number = 2; detecting platform's number = 3; detecting radar's number = 4; MPT number of associated track = 5; updating platform's number

2.3.8 *Subroutine LNKDET*

Subroutine LNKDET is called by the MULSIM executive routine. Its primary purpose is to chronologically order the detections associated with the tracks in the sector track file designated by the sector/platform in the calling sequence. This includes detections made by the updating platform and those provided by other platforms via the communications link.

LNKDET first interrogates the LNKSTO file, which contains all of the detections transmitted over the link in the last 30 s. Those detections associated with tracks from the updating platform (regardless of which sector the track may be in) are selected and stored in the TESTO linked file, which, for each track, contains the detection numbers, the detecting platforms, and the detecting radars corresponding to those detections associated with the track that have not yet been used to update the track. After storing the detections in the TESTO files, the locations that housed the detections in the LNKSTO file are vacated.

The next step in the process is to consider each track in the sector designated in the calling sequence. The detections associated with each track are called up from the TESTO file and their locations in TESTO are placed in the ILOC sequential file according to their times of detection. The location of the detection with the oldest detection time is placed at the top of the file, i.e., ILOC(1). The locations in the TESTO file are vacated in the UPDATE subroutine only after the detections are used in the updating process.

The ordered sequential file (ILOC) is presented to UPDATE for the updating procedure. This process is repeated until all tracks in the designated sector have been considered, whereupon control is returned to the MULSIM executive routine. Table 2.18 defines variables used in LNKDET; Fig. 2.16 shows the flowchart for the subroutine.

Table 2.18 — Variables in LNKDET

Fortran Variable	Description
ISHIP, ISEC, IRAD	Platform, sector, and radar to be considered for updating process
DUMSX(ISEC,ISHIP)	Number of the last track to be placed in platform ISHIP's dummy track file
LNKFSX	Location of last associated detection to be placed in LNKSTO file
LNKSTO	See Table 2.17.
DUMST(KS,MS,IS)	Number of the track platform IS's sector track files that corresponds to track MS from platform KS's MPT files
TRAN(ID,IR,IS)	Time at which platform IS transmitted data on detection ID
TMRK(ISEC,IR,IS)	Sector-crossing time for sector ISEC, radar IR on platform IS
TMS(ID,IR,IS	Time at which platform IS detected detection ID with radar IR
TLAST(NT,IS)	Time at which track NT from platform IS's dummy track file was last updated
TESTO(LOCC,I)	See Table 2.16.
FILFX(NT,ISHIP)	See Table 2.16.
FILID(LOC)	See Table 2.16.
PTFST(NT,IS)	Number of the platform responsible for updating track NT from platform IS's dummy track file

Table 2.18 (Concluded) — Variables in LNKDET

Fortran Variable	Description
TRKST(NT,IS)	Number of the MPT corresponding to dummy track NT
MPFLAG(MT,KS)	Indicator — MPFLAG = 0 indicates that track MT is not a participating platform, otherwise MPFLAG is assigned a value equal to the target number corresponding to the platform
DUMID(NT,IS)	Number of the track that was placed in platform IS's dummy track file prior to track NT

2.3.9 *Subroutine UPDATE*

Subroutine UPDATE is called by subroutine LNKDET to carry out the updating process for NT, a designated track. There are two separate paths through UPDATE. One path is followed when NT is a participating platform with position updates coming from NAVSTAR or JTIDS, and the other is followed when a target is being tracked from radar measurements.

For those tracks that are of participating platforms, an attempt is made to emulate the operation of a system that includes NAVSTAR or JTIDS. Updates to participating platforms are made every 2 s and, to allow for time lags in the system, to within 1 s of current time. Once it has been determined that an update has not been made within the last 2 s, a call is made to SHPGEN and SCOORD to determine the platform's position at current time less 1 s. Because of the relative accuracy of JTIDS and NAVSTAR, the true stabilized coordinates are used as measurements, and the measurement covariance matrix is arbitrarily set with diagonal elements of 100 m^2 and off-diagonal elements of 0.1 m^2. This information is presented to KALMAN for the platform updating.

For those tracks that are not of participating platforms, the first step in the process is to examine the list of associated detections in the TESTO file. To allow for delays in processing, the subroutine considers only those detections that were made before TIMUP. TIMUP is equal to current time less some appropriate delay (TIMLAG). The detections are selected chronologically from the TESTO file (oldest first). The corresponding noisy measurements are called up from the XYZMS file and if the detecting platform is not the updating platform, the measurements are transformed to the updating platform's stabilized coordinate system by the TRANSF subroutine. This transformation is also carried out for the true position coordinates, and the measurement covariance matrix is transformed by the COVLNK subroutine. The measurements, together with the time from the last update, are presented to KALMAN, which determines the new smoothed position for the track. After the last detection is considered, the smoothed state vector, the smoothed covariance matrix, and the predicted covariance matrix are stored in arrays for the next call to update.

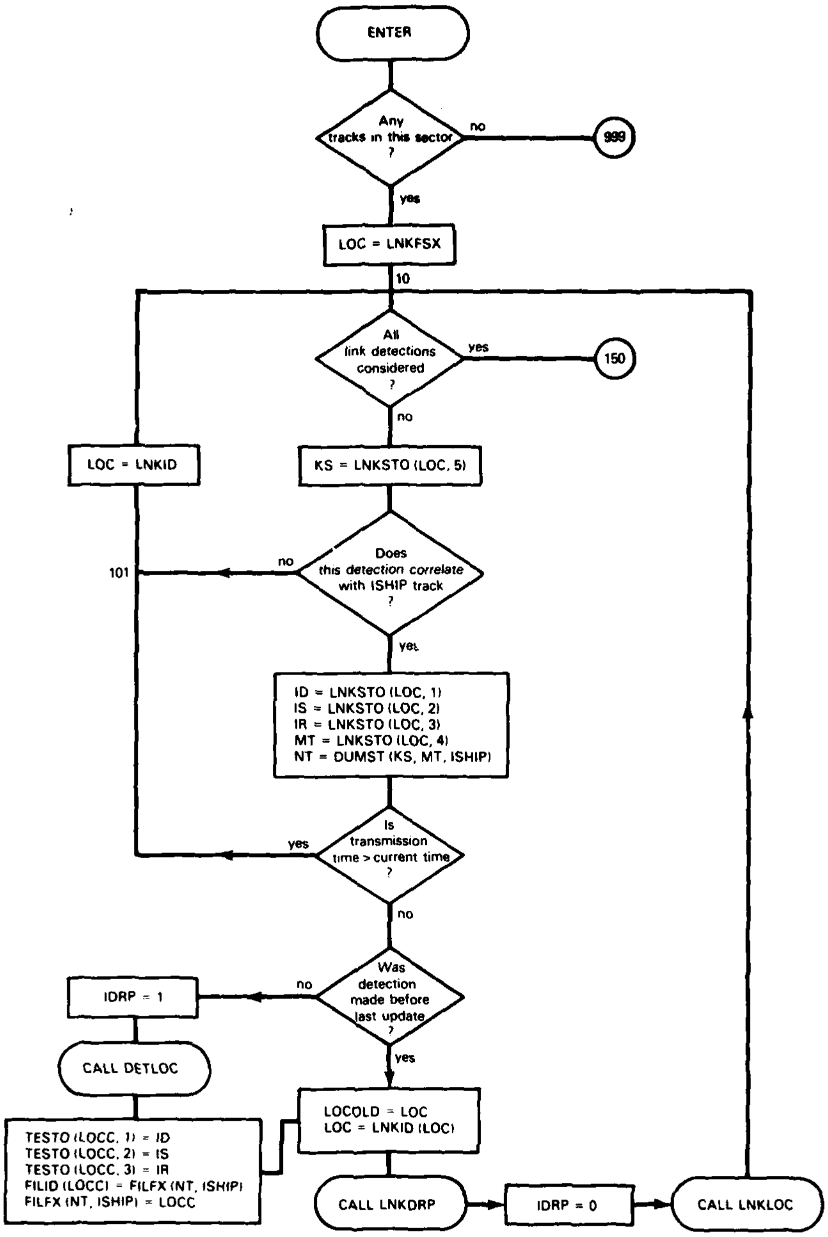

Fig. 2.16 — Subroutine LNKDET

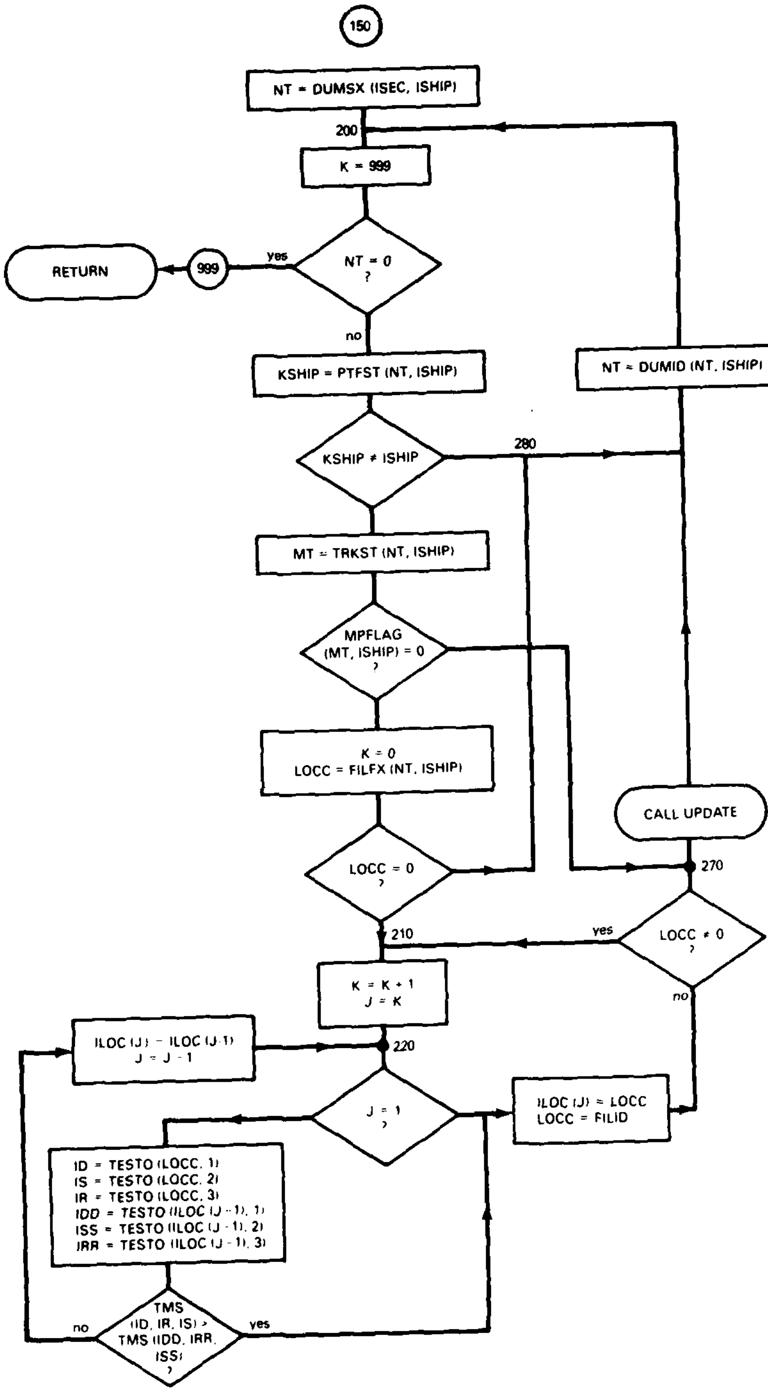

Fig. 2.16 (Concluded) — Subroutine LNKDET

This completes the updating process for track NT, and control is returned to the LNKDET subroutine. A flowchart of the process and a list of Fortran variables are found in Fig. 2.17 and Table 2.19, respectively.

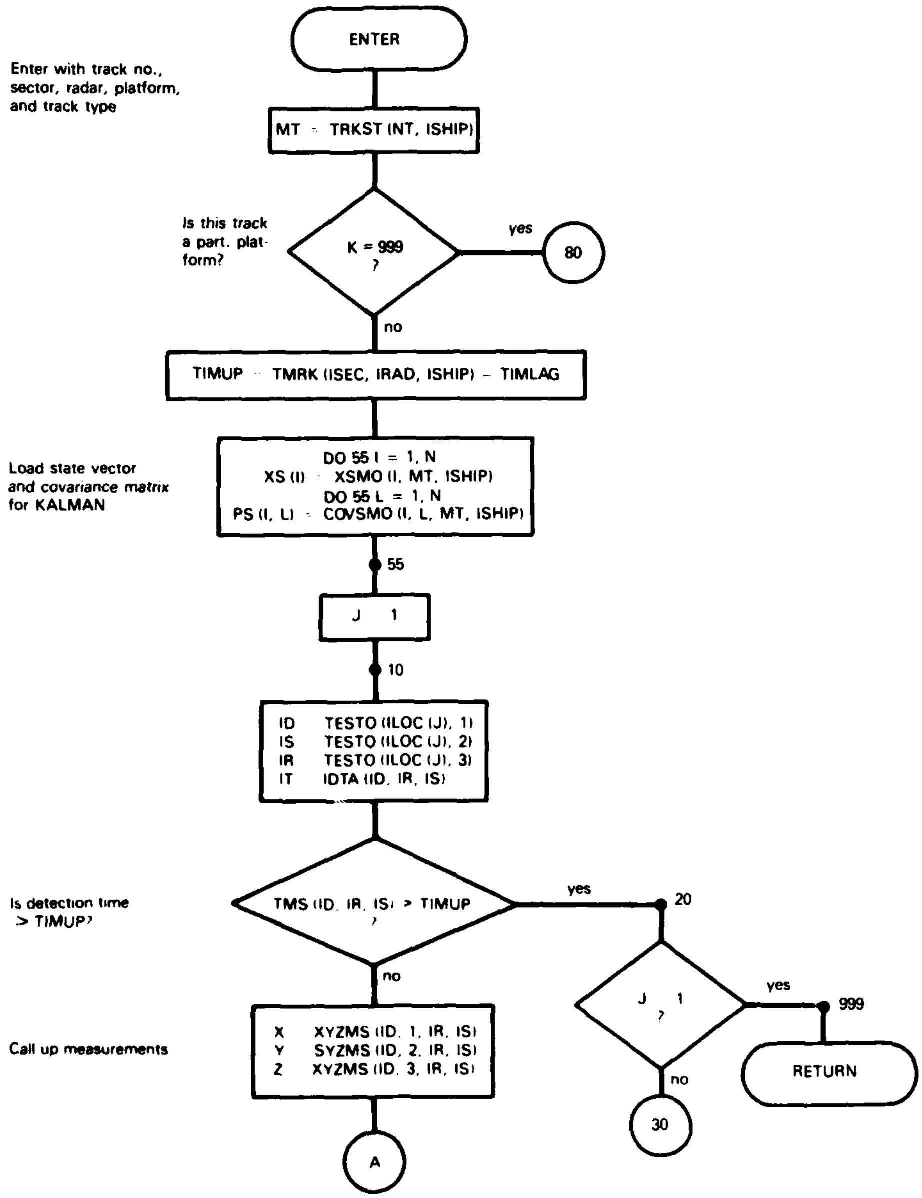

Fig. 2.17 — Subroutine UPDATE

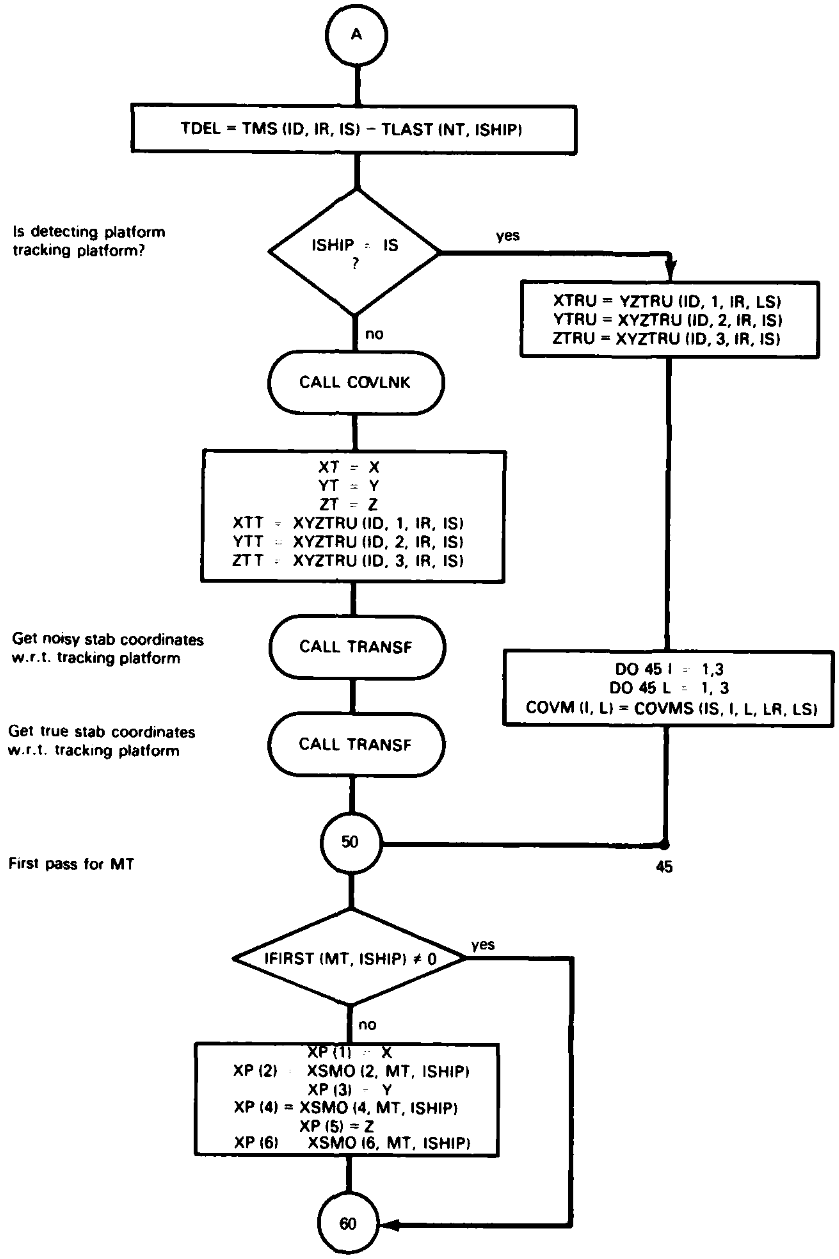

Fig. 2.17 (Continued) — Subroutine UPDATE

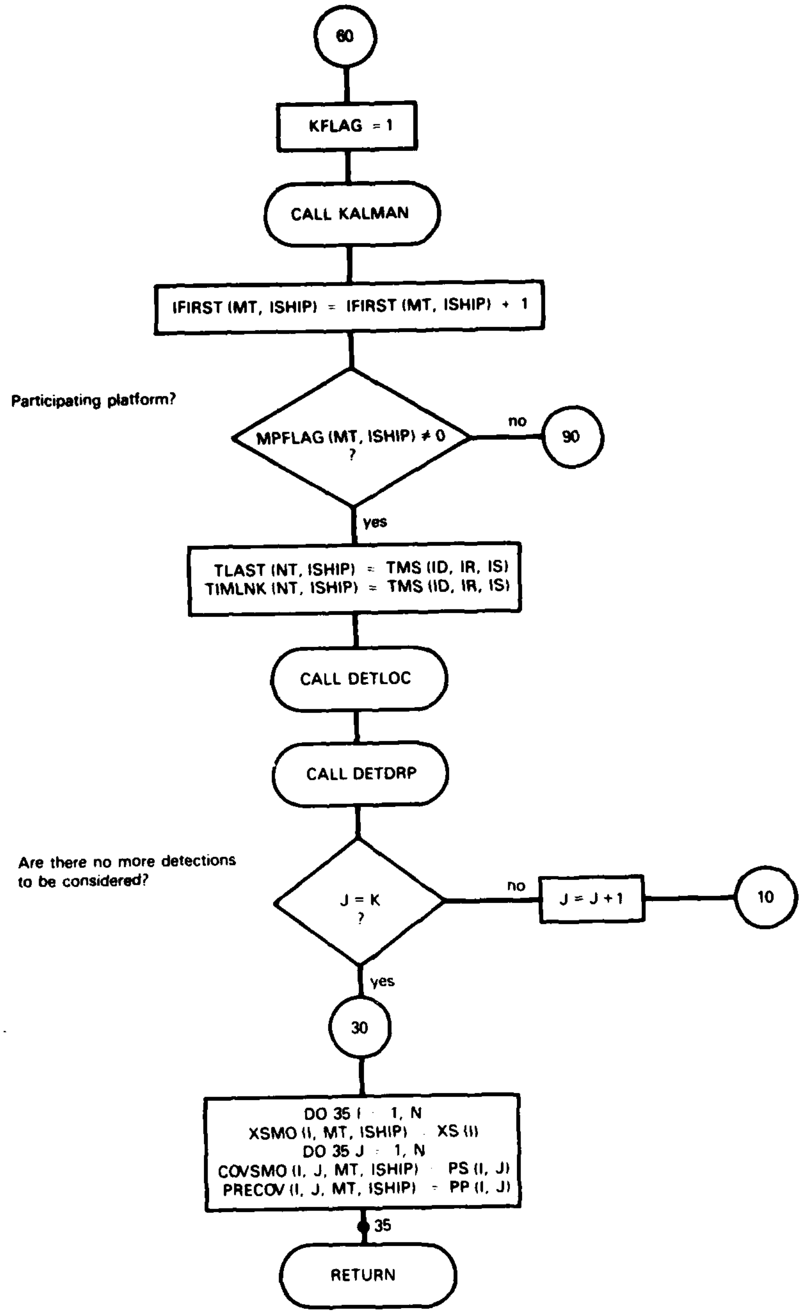

Fig. 2.17 (Continued) — Subroutine UPDATE

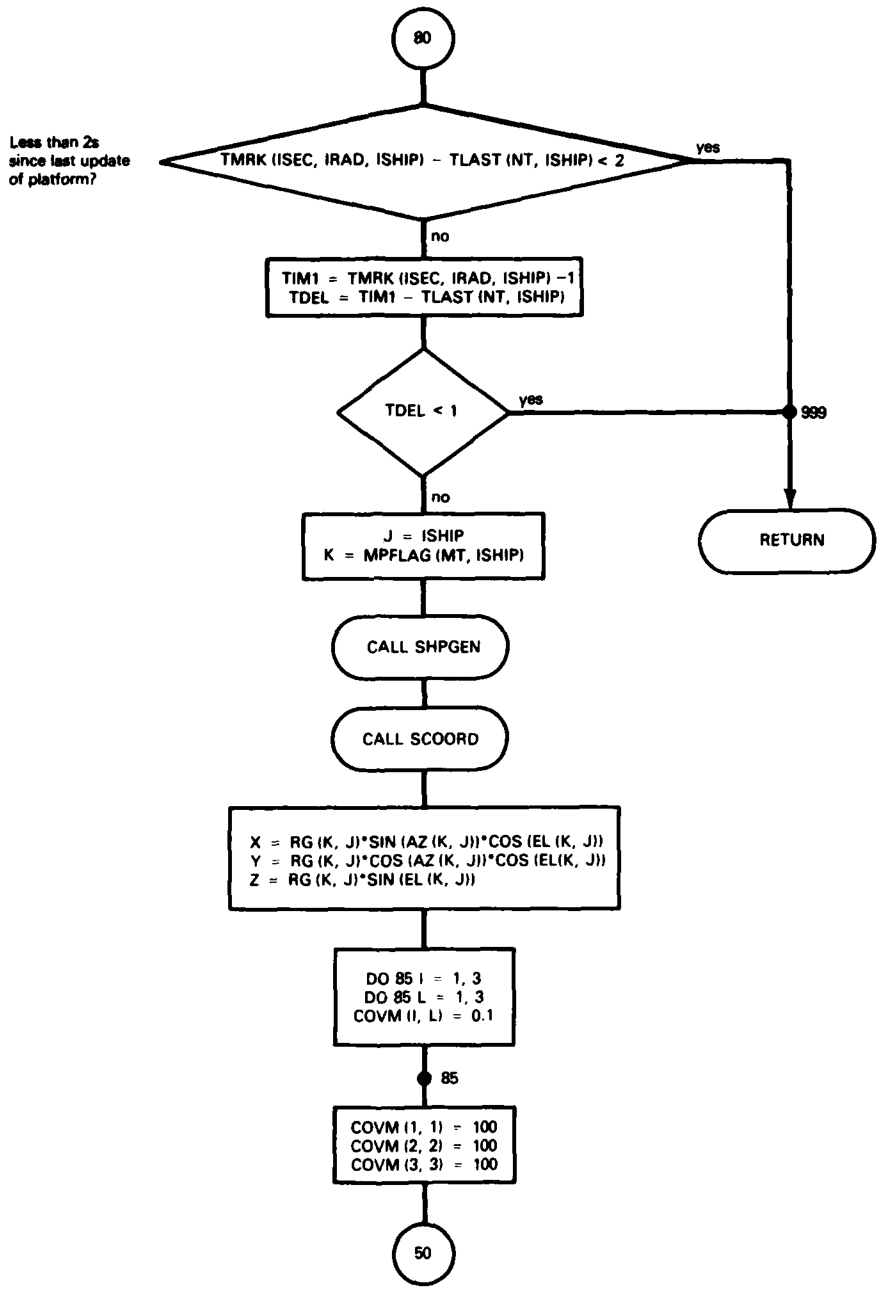

Fig. 2.17 (Concluded) — Subroutine UPDATE

Table 2.19 — Variables in UPDATE

Fortran Variable	Description
NT	Track number of updating candidate.
ISEC, IRAD, ISHIP	Sector containing NT, the radar that detected NT, and the detecting platform
K	Indicator K = 999 indicates that NT is a participating platform; if K ≠ 999, then K equals the number of detections that have been associated with NT.
TRKST(I,J)	File that contains the MPT track number of dummy track I from platform J's dummy track file
TMRK(I,J,K)	Time at which radar J on platform K crosses sector I
TIMLAG	Time lag to allow for processing/transmission delays
TIMUP	Current time less TIMLAG. Track NT will be updated to TIMUP but not beyond.
XS(I)	Smoothed state vector
PS(I,L)	Smoothed covariance matrix
TESTO(ILOC(J),K)	See Table 2.16.
IDTA(ID,IR,IS)	File containing target number of detection ID from radar IR on platform IS
XYZMS(ID,J,IR,IS)	Measurement coordinates of detection ID in platform IS's stabilized coordinate system
TLAST(NT,ISHIP)	Time of last update for track NT
XYZTRU(ID,J,IR,IS)	True position coordinates of detection ID in platform IS's stabilized coordinate system
XSMØ(I,MT,IS)	Smoothed state vector of track MT from platform IS's MPT file
TMS(ID,IR,IS)	Time at which detection ID was detected by radar IR on platform IS
MPFLAG	See Table 2.18.
IFIRST(MT,IS)	Counter that records the number of calls to UPDATE for track MT
RG(I,J),AZ(I,J) EL(I,J)	Range, azimuth, and elevation of target I with respect to platform J in platform J's stabilized coordinate system
COVM(I,J)	Measurement covariance matrix
COVSMO(I,J,MT,IS)	Error covariance for track MT
PRECOV(I,J,MT,IS)	Predicted covariance for track MT

3.0 SUMMARY AND RESULTS

A computer simulation has been developed that will serve as a foundation for future software development and at the same time allow the user to demonstrate the advantages and limitations inherent in a multiple platform sensor integration system.

Some preliminary results have been generated through the use of a simplified scenario used in program development (see Fig. 3.1). In the scenario three ships are moving due east near the equator. They are separated by 1° in latitude (60 n.mi.) or longitude, and a single target is approaching from the east at 1000 m/s. Each ship has a single radar. The measurement accuracies of all three radars are modeled by assuming standard deviations of 100 m on the range measurement, 0.5° in the azimuth measurement, and 1° in the elevation measurement. The radar on ship 1 has a 4-s scan rate and the radars on ships 2 and 3 are scanning at a 6-s rate.

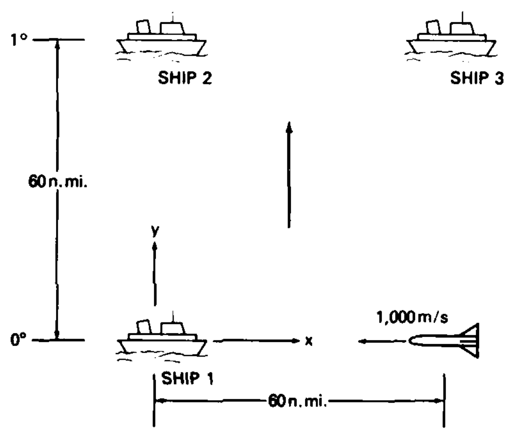

Fig. 3.1 — Simplified scenario (initial positions)

An example of a single ship's tracking capability over a 100-s time interval is shown in Fig. 3.2. In this case the measured and smoothed y coordinate of the target as determined by ship 1 is presented. Since the y measurement as determined by ship 1 is essentially an angular measurement, it does not have the accuracy associated with a range measurement; consequently the measurement tends to be quite noisy.

When measurements from ship 2 are combined with measurements from ship 1 to produce the smoothed y position, there is significant improvement in the system's tracking accuracy, as shown in Fig. 3.3.

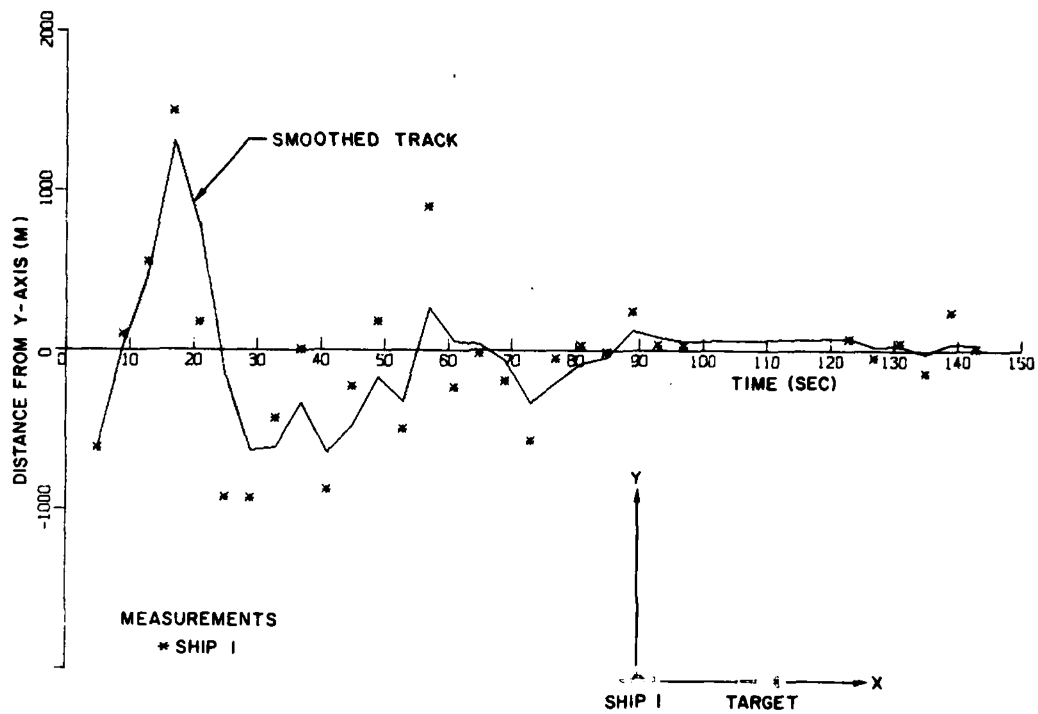

Fig. 3.2 — One-ship system; track of target's y coordinate

The improvement is less pronounced when ship 3 makes its contribution. However, it should be noted that initially the y coordinate as determined by ship 3 is essentially a range measurement that is very accurate. This is apparent in Fig. 3.4 as evidenced by the large number of detections from ship 3, which initially lie close to the true position. These accurate detections, together with the increased data rate, enable the 3-ship system to produce a highly accurate track in a relatively short time. The accuracy of ship 3's y measurement decreases with time, whereas that of ship 2's improves as the target approaches ship 1.

Results have also been generated with a rudimentary contact selector that rejects track measurements when a previous measurement has been made within some designated time slot. Time slots of 3 and 5 s have been used with a 3-ship system to generate the results shown in Figs. 3.5 and 3.6. The results are encouraging. The 3-s time slot, with a 25% reduction in the number of detections, shows little degradation of track quality, and the 5-s time slot with a 43% reduction gives track quality comparable to that of a 2-ship system with no contact selector.

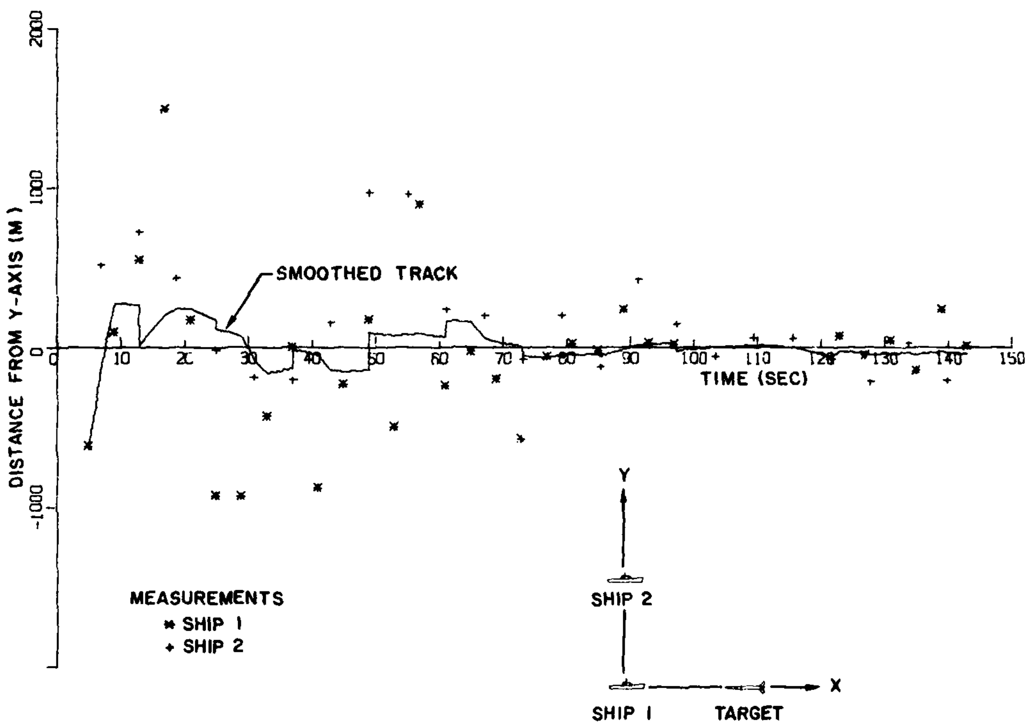

Fig. 3.3 — Two-ship system; track of target's y coordinate

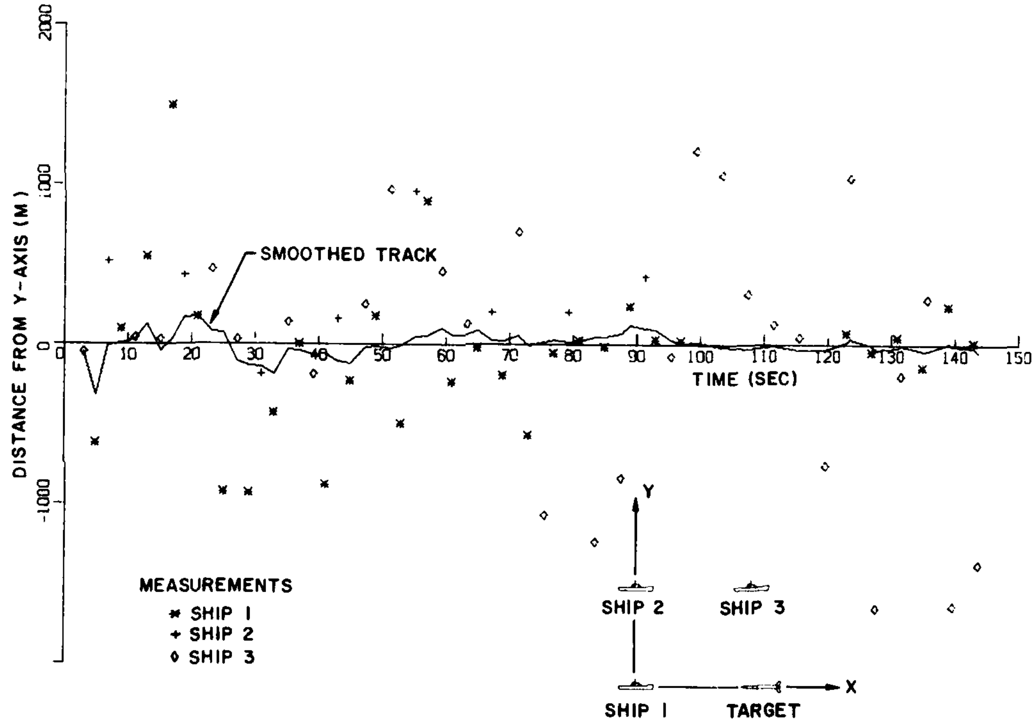

Fig. 3.4 — Three-ship system; no contact selector. Track of target's y coordinate

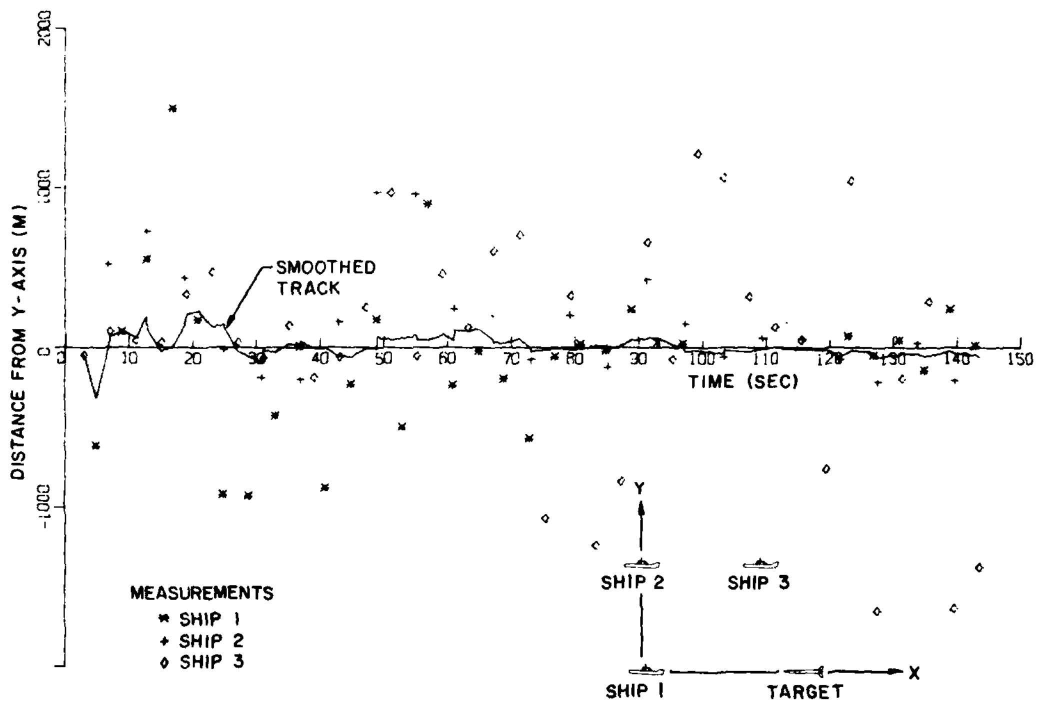

Fig. 3.5 — Three-ship system with 3-s time slot; track of target's *y* coordinate

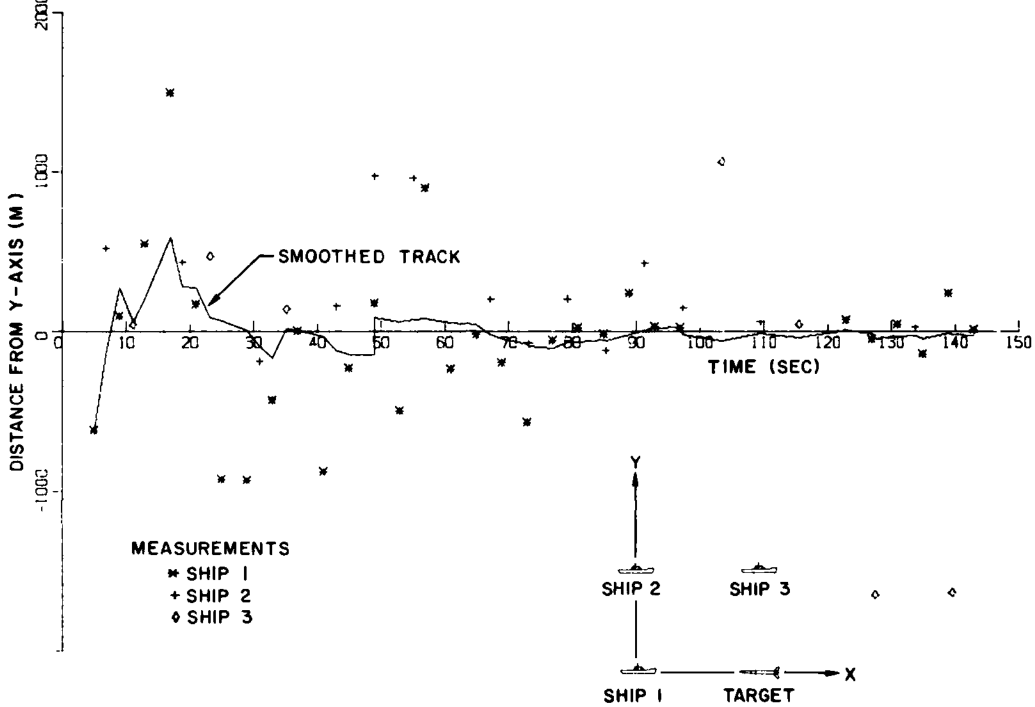

Fig. 3.6 — Three-ship system with 5-s time slot; track of target's *y* coordinate

4.0 ACKNOWLEDGMENT

The author wishes to thank Dr. B. H. Cantrell for fruitful discussions and suggestions relating to many aspects of this report.

5.0 REFERENCES

1. B.H. Cantrell and A. Grindlay, "Multiple Platform Radar Tracking System," paper to be presented at the International Radar Conference, Apr. 1980.
2. B.H. Cantrell, A. Grindlay and C.H. Dodge, "Formulation of a Platform-to-Platform Radar Integration System," NRL Memorandum Report 3404, Dec. 1976.
3. G.V. Trunk and J.D. Wilson, "Tracking Filters for Multiple-Platform Radar Integration," NRL Report 8087, Dec. 14, 1976.
4. A. Grindlay, "Modeling a Multiple Platform Sensor Integration System," paper presented at ORSA National Meeting, Los Angeles, CA, Nov. 1978.

Appendix

PROGRAM LISTINGS

```
SOURCE LISTING

  STATEMENT



  PROGRAM MAIN
  COMMON/PLOT/IPLT(1000),XX1(1000),YYY(1000),NP,YYN(1000),IYIS(1000)
  COMMON/MODULO/ISLOT(20,5,60),IKEY(3),IMOD20,IMOD60
  COMMON/RADNEX/SECTIM(3,5),TIMNEX(3,5),NEXSEC(3,5),NR(5),NS,NT
  COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
  COMMON/DATEUP/NUMTRG,NUMSHP
  COMMON/SECTOR/KSEC(64,5)
  CALL RSSTOP
  CALL PLOTS(IPLT,1000,0,55)
  NP = 1
  PRINT 300
300 FORMAT(/,15X,'TIME',7X,'I. D.',19X,'X',12X,'Y',12X,'Z',11X,'VX',
 111X,'VY',11X,'VZ',7X,'P ATFORM')
  TIME = 0.
  KNIT = 1
  CALL INITAL
  NUMSHP = NS
  NUMTRG = NT
  NTS = NT+NS
10 CONTINUE
  CALL TRKGEN(TIME,NT)
  CALL SHPGEN(TIME,NS)
  CALL MOTION(TIME,NS)
  DO 20 J=1,NS
  CALL SCOORD(NT,NS,J)
  CALL TCOORD(NT,J)
  CALL STAB1(NT,NS,J)
20 CONTINUE
  CALL LOAD(KNIT,TIME)
  IF(KNIT.EQ.2) GO TO 30
  KNIT = 2
  TIME = 1
  GO TO 10
30 CONTINUE
  CALL NEXRAD(IR,ISHIP,ISEC,TIME)
  IF(KSEC(ISEC,ISHIP).NE.0) GO TO 32
  IF(KSEC(ISEC-1,ISHIP).NE.0) GO TO 32
  IF(KSEC(ISEC+1,ISHIP).NE.0) GO TO 32
  IF(KSEC(ISEC-2,ISHIP).NE.0) GO TO 32
  IF(KSEC(ISEC+2,ISHIP).NE.0) GO TO 32
  GO TO 38
32 CONTINUE
  DO 35 I=1,64
  KSEC(I,ISHIP) = 0
35 CONTINUE
  CALL TRKGEN(TIME,NT)
  CALL SHPGEN(TIME,NS)
  CALL MOTION(TIME,NS)
```

MAIN

CSN

```
      CALL SCOORD(NT,NS,ISHIP)
      CALL TCOORD(NT,ISHIP)
      CALL DETFIL(IR,ISHIP,ISEC,NT,NS)
      CALL STAB1(NT,NS,ISHIP)
      CALL NOISY(IR,ISHIP,ISEC)
      IF(TIME.LE. 1.1) GO TO 30
   38 CONTINUE
      ISEC = ISEC-1
      IF(ISEC.EQ.0) ISEC=64
      CALL PREDIC(ISEC,IR,ISHIP)
      ISEC = ISEC-1
      IF(ISEC.EQ.0) ISEC=64
      CALL CORRAS(ISHIP,ISEC,IR)
      CALL SORT(ISEC,ISHIP,IR)
      ITIM = TIME
      MOD20  = MOD(ITIM,IMOD20)
      IF(MOD20.NE.0) GO TO 60
      MOD60 = MOD(ITIM,IMOD60)
      IDEM = (MOD60/20) + 1
      IF(IKEY(IDEM).NE.0) GO TO 60
      K = MOD60+ 20
      J = MOD60+1
      DO 55 I=J,K
      DO 55 IS=1,5
      DO 55 MT=1,20
      ISLOT(MT,IS,I) = 0
   55 CONTINUE
      IKEY(IDEM) = 1
      IDEM = IDEM+1
      IF(IDEM.EQ.4) IDEM=1
      IKEY(IDEM) = 0
   60 CONTINUE
      CALL LNKDET(ISHIP,ISEC,IR)
      IF(TIME.LT.150.) GO TO 30
      NP = NP-1
      RETURN
      END
```

CSN

```
      SUBROUTINE INITAL
      COMMON/LOCNEW/LASTO,FULSTO,LISTO(256),NEXSTO,KJ
      COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
      COMMON/TAG/ITAG(20,3,5,64)
      COMMON/COV1/ SIGAZD(20,2),SIGELD(20,2),RHOD(20,2)
      COMMON/STATIC/N(9),N2(3)
      COMMON/KAL4/IFIRST(20,5),DIM1,DIM2,DIM3
      COMMON/MPT/DROPM(5),MPTNO(5),FULLM(5),LISTM(5,20),
     1NEXTM(5),LASTM(5)
      COMMON/LOCDET/LASDET,FULDET,LISDET(256),NEXDET
      COMMON/LOAD/XSAV(3,20,3),AZINT(3,5),RVEL(3,5),RNGDIM(5,3)
      COMMON/DUM/DROPD(5),DUMNO(5),FULLD(5),LISTD(5,512),NEXTD(5),
     1LASTD(5)
      COMMON/CORAS/DETSX(256),ISTOR(256),DETID(512)
      COMMON/SORT/FILFX(256,5),FILID(512)
      COMMON/RADNEX/SECTIM(3,5),TIMNEX(3,5),NEXSEC(3,5),NR(5),NS,NT
      COMMON/PAR3/TILAT(20),TILOG(20),TIHT(20),TIV(20),TIHD(20)
      COMMON/PAR4/SILAT(20),SILOG(20),SIHT(20),SIV(20),SIHD(20)
      COMMON/PAR8/ RMAG(20),PMAG(20),WRR(20),WRP(20)
      COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/LOCLNK/LASLNK,FULLNK,LISLNK(20),NEXLNK
      COMMON/KAL1/ PHI(20,20),G(20,20),H(20,20)
      COMMON/MODULO/ISLOT(20,5,60),IKEY(3),IMOD20,IMOD60
      COMMON/RANDUM/IRAN(20,5)
      DIMENSION TVEL(256),SVEL(5)
      REAL N,N2
      INTEGER FILID,FILFX,FULLM,DROPM,FULLD,DROPD,DIM1,DIM2,DIM3,FULLNK
      INTEGER DETSX,DETID,FULSTO
      CALL SETVR(17)
      KJ = 0
      RAD=.01745329252
      DIM1 = 3
      DIM2 = 6
      DIM3 = 0
      TIMLAG = 2
      LASDET = 256
      NEXDET = 1
      RNGDIM(1,1) = 10000.
      RNGDIM(2,1) = 10000.
      N(1) = 15.
      N(2) = 15.
      N(3) = 20.
      N(4) = 100.
      N(5) = 1.
      N(6) = .5
      N(7) = 15.
      N(8) = 15.
      N(9) = 20.
      N2(1) = 100.
```

INITAL

CSN

```
      N2(2) = 1.
      N2(3) = .5
      N(5) = N(5)*RAD
      N(6) = N(6)*RAD
      N2(2) = N2(2)*RAD
      N2(3) = N2(3)*RAD
      SIGAZD(2,1) = .5
      SIGELD(2,1) = 1.0
      RHOD(2,1) = 100.
      SIGAZD(1,1)=.5
      SIGELD(1,1)=1.0
      RHOD(1,1)=100.
      DO 10 I=1,255
      LISOET(I) = I+1
      DETSX(I) = 0
   10 CONTINUE
      DO 20 I=1,256
      DO 20 J=1,5
      FILFX(I,J) = 0
   20 CONTINUE
      DO 30 I=1,512
      FILID(I) = 0
   30 CONTINUE
      DO 200 I=1,20
      DO 200 J=1,5
      IRAN(I,J) = J+5*(I-1)
  200 CONTINUE
      NS = 1
      NS = 2
      NS = 3
      SIGAZD(3,1) = .5
      SIGELD(3,1) = 1.0
      RHOD(3,1) = 100.
      AZINT(1,3) = 270.
      RVEL(1,3) = 90.
      SILAT(3) = 1.
      SILOG(3) = 1.
      SIHT(3) = 0.
      SIHD(3) = 90.
      SVEL(3) = 10.
      RNGDIM(3,1) = 10000.
      NR(3) = 1
      NT = 1
      NR(1) = 1
         NR(2) = 1
      AZINT(1,2) = 90.
      AZINT(1,1) = 0
      RVEL(1,1) = 90.
      RVEL(1,2) = 60.
      DO 40 J=1,NS
```

INITAL

CSN

```
      K = NR(J)
      DO 40 I=1,K
      SECTIM(I,J) = 5.625/RVEL(I,J)
   40 CONTINUE
      SILAT(1)=0.
      SILOG(1)=0.
      SIHT(1)=0.
      SIHD(1)=90.
      TILAT(1) = 0.1
      TILAT(1)=0.
      TILOG(1)=1.
      TIHT(1)=2000.
      TIHD(1)=-90.
      SILAT(2)=1.
      SILOG(2)=0.
      SIHT(2)=0.
      SIHD(2)=90.
      TVEL(1)=300.
      TVEL(1)=900.
      TVEL(1) = 1000.
      SVEL(1)=10.
      SVEL(2)=10.
      ER=6378388.
      PR=6359911.
      Q1 = ER*ER
      Q2=PR*PR
      DO 50 I=1,NT
      SI=SIN(TILAT(I)*RAD)
      CO=COS(TILAT(I)*RAD)
      RHOT=SQRT(Q1*Q2/(Q1*SI*SI+Q2*CO*CO))
      RHOT=RHOT+TIHT(I)
      TIV(I)=(TVEL(I)/RHOT)/RAD
   50 CONTINUE
      DO 60 I=1,NS
      SI=SIN(SILAT(I)*RAD)
      CO=COS(SILAT(I)*RAD)
      RHOS=SQRT(Q1*Q2/(Q1*SI*SI+Q2*CO*CO))
      RHOS=RHOS+SIHT(I)
      SIV(I)=(SVEL(I)/RHOS)/RAD
   60 CONTINUE
C MOTION INPUTS
      RMAG(1)=0.
      PMAG(1)=0.
      WOR(1)=0.
      WOP(1)=0.
      RPHASE(1)=0.
      PPHASE(1)=0.
      RMAG(2)=0.
      PMAG(2)=0.
      WOR(2)=0.
```

INITAL

CSN

```
            WBP(2)=0.
            RPHASE(2)=0.
            PPHASE(2)=0.
      C MPTFIL AND DUMFIL INITIALIZATION
            DO 70 I=1,5
            LASTM(I) =20
            DROPM(I) =1
            FULLM(I) = 19
            LASTM(I) = 20
            NEXTM(I) =1
            DO 70 J=1,19
            LISTM(I,J) = J+1
         70 CONTINUE
            DO 80 I=1,5
            DROPD(I) = 1
            LASTD(I) = 512
            FULLD(I) = 511
            NEXTD(I) = 1
            DO 80 J=1,511
            LISTD(I,J) = J+1
         80 CONTINUE
            FULLNK = 19
            LASLNK = 20
            NEXLNK = 1
            DO 90 I=1,19
            LISLNK(I) = I+1
         90 CONTINUE
            DO 110 I=1,DIM2
            DO 110 J=1,DIM1
            G(I,J) = 0.
            H(J,I) = 0.
        110 CONTINUE
            H(1,1) = 1.
            H(2,3) = 1.
            H(3,5) = 1.
            IMOD20 = 20
            IMOD60 = 60
            DO 120 I=1,20
            DO 120 J=1,5
            DO 120 K=1,60
            ISLOT(I,J,K) = 0
        120 CONTINUE
            DO 130 IT=1,20
            DO 130 IR=1,3
            DO 130 IS=1,5
            DO 130 ISEC=1,64
            ITAG(IT,IR,IS,ISEC) = 0
        130 CONTINUE
            NEXSTO = 1
            FULSTO = 255
            LASTO = 256
            DO 140 I=1,255
            LISTO(I) =I+1
        140 CONTINUE
            RETURN
            END
```

```
ISN

0001          SUBROUTINE LOAD(KNIT,TIME)
0002          COMMON/KAL2/ PS(20,20),PP(20,20),COVS(20,20),COVM(20,20),XP(20)
0003          COMMON/LOAD/XSAV(3,20,3),AZINT(3,5),RVEL(3,5),RNGDIM(5,3)
0004          COMMON/RADNEX/SECTIM(3,5),TIMNEX(3,5),NEXSEC(3,5),NR(5),NS,NT
0005          COMMON/MPT/DROPM(5),MPTNO(5),FULLM(5),LISTM(5,20),
             1NEXTM(5),LASTM(5)
0006          COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
0007          COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
             10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
             2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
0008          COMMON/DUMSEC/DUMSX(64,5),DUMID(512,5)
0009          COMMON/DUM/DROPD(5),DUMNO(5),FULLD(5),LISTD(5,512),NEXTD(5),
             1LASTD(5)
0010          COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
0011          COMMON/LODE/NUMTAR
0012          INTEGER DUMSX,DUMID,FULLD,DROPD,DUMNO,PTFST,TRKST,DROPM,DUMST
0013          NTS = NT+NS
0014          DO 100 J=1,NTS
0015          DO 100 K=1,NS
0016          IF(KNIT.EQ.2) GO TO 40
0017          IF((J-NT).EQ.K) GO TO 100
0018          XSAV(1,J,K) = RG(J,K)*SIN(AZ(J,K)) * COS(EL(J,K))
0019          XSAV(2,J,K) = RG(J,K)*COS(AZ(J,K)) * COS(EL(J,K))
0020          XSAV(3,J,K) = RG(J,K)*SIN(EL(J,K))
0021          GO TO 100
0022       40 CONTINUE
0023          IF((J-NT).EQ.K) GO TO 100
0024          JSEC = AZ(J,K)*10.185916
0025          ISEC = JSEC + 1
0026          DROPD(K) = 1
0027          CALL DUMFIL(K)
0028          CALL DUMNEW(DUMNO(K),ISEC,K)
0029          IF(K.EQ.1) GO TO 45
0030          IF((J-NT).NE.1 .OR.K.NE.2) GO TO 50
0031       45 CONTINUE
0032          DROPM(K) = 1
0033          CALL MPTFIL(K)
0034          TRKST(DUMNO(K),K) = MPTNO(K)
0035          PTFST(DUMNO(K),K) = K
0036          DUMST(K,MPTNO(K),K) = DUMNO(K)
0037          XSMO(1,MPTNO(K),K) = RG(J,K)*SIN(AZ(J,K)) * COS(EL(J,K))
0038          XSMO(3,MPTNO(K),K) = RG(J,K)*COS(AZ(J,K)) * COS(EL(J,K))
0039             XSMO(5,MPTNO(K),K) = RG(J,K)*SIN(EL(J,K))
0040          XSMO(2,MPTNO(K),K) = (XSMO(1,MPTNO(K),K)-XSAV(1,J,K))/TIME
0041          XSMO(4,MPTNO(K),K) = (XSMO(3,MPTNO(K),K)-XSAV(2,J,K))/TIME
0042          XSMO(6,MPTNO(K),K) = (XSMO(5,MPTNO(K),K)-XSAV(3,J,K))/TIME
0043          TLAST(DUMNO(K),K) = 1.
0044          TIMLNK(MPTNO(K),K) = 1
        C INITIALIZE COVARIANCE FOR FILTER
0045          IRD =1
```

LOAD

CSN

```
      IDT = 0
      CALL STAB2(J,K,IRD)
      NUMTAR = J
      CALL COVOWN(IRD,K,IDT,DUMNO(K),SD,ISEC)
      COVSMO(1,1,MPTNO(K),K) = COVM(1,1)
      COVSMO(3,3,MPTNO(K),K) = COVM(2,2)
      COVSMO(5,5,MPTNO(K),K) = COVM(3,3)
      COVSMO(1,3,MPTNO(K),K) = COVM(1,2)
      COVSMO(3,1,MPTNO(K),K) = COVM(1,2)
      COVSMO(1,5,MPTNO(K),K) = COVM(1,3)
      COVSMO(5,1,MPTNO(K),K) = COVM(1,3)
      COVSMO(3,5,MPTNO(K),K) = COVM(2,3)
      COVSMO(5,3,MPTNO(K),K) = COVM(2,3)
      MPFLAG(MPTNO(K),K) = 0
      IF(J.GT.NT) MPFLAG(MPTNO(K),K) = J
C MPFLAG NE ZERO INDINICATES PARTICIPATING PLATFORM
      GO TO 100
   50 CONTINUE
      TLAST(DUMNO(K),K) = 1.
      IF(K.GT.2 .AND. (J-NT).EQ.1) GO TO 55
      TRKST(DUMNO(K),K) = MPTNO(1)
      PTFST(DUMNO(K),K) = 1
      DUMST(1,MPTNO(1),K) = DUMNO(K)
      GO TO 100
   55 CONTINUE
      TRKST(DUMNO(K),K) = MPTNO(2)
      PTFST(DUMNO(K),K) = 2
      DUMST(2,MPTNO(2),K) = DUMNO(K)
  100 CONTINUE
      DO 60 J=1,NS
      K = NR(J)
      DO 60 I=1,K
      AZRAD = AZINT(I,J) + TIME*RVEL(I,J)
      NEXSEC(I,J) = AZRAD/5.625
      RSEC  = NEXSEC(I,J) + 1.
      AZTO = (RSEC*5.625) - AZRAD
      TIMNEX(I,J) = AZTO/RVEL(I,J)+TIME
      NEXSEC(I,J) = NEXSEC(I,J) + 1
      IF(NEXSEC(I,J).GT.64) NEXSEC(I,J) = NEXSEC(I,J)-64
   60 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE NEXRAD(IR,IS,ISEC,TIME)
C***********************************************************************
C SUBROUTINE NEXRAD DETERMINES WHICH RADAR WILL NEXT MAKE A SECTOR CROSS-
C ING. ITS OUTPUT IDENTIFIES THE RADAR,THE SHIP, AND THE SECTOR. THE TIME
C OF THE SECTOR CROSSING IS ALSO GIVEN.
C***********************************************************************
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/RADNEX/SECTIM(3,5),TIMNEX(3,5),NEXSEC(3,5),NR(5),NS,NT
   10 CONTINUE
      K=1
      L=1
      SAVTIM = 100000.
      DO 20 J=1,NS
      JNR = NR(J)
      DO 20 I=1,JNR

C CHECK TO SEE IF TIME OF NEXT SECTOR CROSSING FOR THIS RADAR IS GREATER
C THAN SAVTIM.

      IF(TIMNEX(I,J).GT.SAVTIM) GO TO 20
      K = I
      L = J
      SAVTIM = TIMNEX(I,J)
   20 CONTINUE
      ISEC = NEXSEC(K,L)
      IR = K
      IS = L
      NEXSEC(K,L) = NEXSEC(K,L)+1
      IF(NEXSEC(K,L).LT.65) GO TO 30
      NEXSEC(K,L) = 1
   30 CONTINUE
      TMRK(ISEC,K,L) = TIMNEX(K,L)
      TIMNEX(K,L) =  TIMNEX(K,L) + SECTIM(K,L)
      IF(TMRK(ISEC,K,L).LT.TIME) GO TO 10
      TIME = SAVTIM
      RETURN
      END
```

CSN

```
      SUBROUTINE MOTION(TIME,NS)
      COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/PAR8/ RMAG(20),PMAG(20),WOR(20),WOP(20)
      RAD=.01745329252
      DO 100 I=1,NS
      ARG=COS((WOR(I)*TIME+RPHASE(I))*RAD)
      ROLL(I)=RMAG(I)*ARG
      ARG=COS((WOP(I)*TIME+PPHASE(I))*RAD)
      PITCH(I)=PMAG(I)*ARG
  100 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE SHPGEN(TIME,NS)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/PAR4/SILAT(20),SILOG(20),SIHT(20),SIV(20),SIHD(20)
      RAD=.01745329252
      DO 500 I=1,NS
      C=SIV(I)*TIME*RAD
      XP=0.
      YP=SIN(C)
      ZP=COS(C)
      X1=YP*SIN(SIHD(I)*RAD)
      Y1=YP*COS(SIHD(I)*RAD)
      Z1=ZP
      CLA=COS(SILAT(I)*RAD)
      SLA=SIN(SILAT(I)*RAD)
      CLO=COS(SILOG(I)*RAD)
      SLO=SIN(SILOG(I)*RAD)
      XG=CLO*X1-SLA*SLO*Y1+CLA*SLO*Z1
      YG=CLA*Y1+SLA*Z1
      ZG=-SLO*X1-SLA*CLO*Y1+CLA*CLO*Z1
      RO=XG*XG+ZG*ZG
      R1=SQRT(RO)
      IF(R1-.000001) 310,340,340
  310 IF(YG) 320,330,330
  320 SLAT(I)=-90.
      GO TO 350
  330 SLAT(I)=90.
      GO TO 350
  340 SLAT(I)=ATAN2(YG,R1)/RAD
  350 AZG=ABS(ZG)
      IF(AZG-.000001) 360,390,390
  360 IF(XG) 370,380,380
  370 SLOG(I)=-90.
      GO TO 400
  380 SLOG(I)=90.
      GO TO 400
  390 SLOG(I)=ATAN2(XG,ZG)/RAD
  400 CONTINUE
      SHT(I)=SIHT(I)
      XP=1.
      YP=0.
      ZP=0.
      AINT=ABS(SIHD(I))-180.
      IF(AINT-.000001) 402,403,403
  402 AINT=180.
  403 AINT=SIHD(I)
      X1=XP*COS(AINT*RAD)
      Y1=-XP*SIN(AINT*RAD)
      Z1=ZP
      CLA=COS(SILAT(I)*RAD)
```

SHPGEN

CSN

```
      SLA=SIN(SILAT(I)*RAD)
      CLO=COS(SILOG(I)*RAD)
      SLO=SIN(SILOG(I)*RAD)
      XG2=CLO*X1-SLA*SLO*Y1+CLA*SLO*Z1
      YG2=CLA*Y1+SLA*Z1
      ZG2=-SLO*X1-SLA*CLO*Y1+CLA*CLO*Z1
      CLO=COS(SLOG(I)*RAD)
      SLO=SIN(SLOG(I)*RAD)
      XG1=CLO
      ZG1=-SLO
      ARG=XG1*XG2+ZG1*ZG2
      IF(ARG-1.) 420,410,410
  410 ARG=1.
  420 IF(ARG+1.) 430,430,440
  430 ARG=-1.
  440 SHD(I)=ACOS(ARG)/RAD
      IF(AINT) 450,460,460
  450 SHD(I)=-SHD(I)
  460 CONTINUE
      IF(SHD(I)) 470,480,480
  470 SHD(I)=360.+SHD(I)
  480 CONTINUE
  500 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE TRKGEN(TIME,NT)
      COMMON/PAR1/TLAT(20),TLOG(20),THT(20),THD(20)
      COMMON/PAR3/TILAT(20),TILOG(20),TIHT(20),TIV(20),TIHD(20)
      RAD=.01745329252
      DO 200 I=1,NT
      C=TIV(I)*TIME*RAD
      XP=0.
      YP=SIN(C)
      ZP=COS(C)
      X1=YP*SIN(TIHD(I)*RAD)
      Y1=YP*COS(TIHD(I)*RAD)
      Z1=ZP
      CLA=COS(TILAT(I)*RAD)
      SLA=SIN(TILAT(I)*RAD)
      CLO=COS(TILOG(I)*RAD)
      SLO=SIN(TILOG(I)*RAD)
      XG=CLO*X1-SLA*SLO*Y1+CLA*SLO*Z1
      YG=CLA*Y1+SLA*Z1
      ZG=-SLO*X1-SLA*CLO*Y1+CLA*CLO*Z1
      RO=XG*XG+ZG*ZG
      R1=SQRT(RO)
      IF(R1-.000001) 10,40,40
   10 IF(YG) 20,30,30
   20 TLAT(I)=-90.
      GO TO 50
   30 TLAT(I)=90.
      GO TO 50
   40 TLAT(I)=ATAN2(YG,R1)/RAD
   50 AZG=ABS(ZG)
      IF(AZG-.000001) 60,90,90
   60 IF(XG) 70,80,80
   70 TLOG(I)=-90.
      GO TO 100
   80 TLOG(I)=90.
      GO TO 100
   90 TLOG(I)=ATAN2(XG,ZG)/RAD
  100 CONTINUE
      THT(I)=TIHT(I)
      XP=1.
      YP=0.
      ZP=0.
      AINT=ABS(TIHD(I))-180.
      IF(AINT-.000001) 102,103,103
  102 AINT=180.
  103 AINT=TIHD(I)
      X1=XP*COS(AINT*RAD)
      Y1=-XP*SIN(AINT*RAD)
      Z1=ZP
      CLA=COS(TILAT(I)*RAD)
```

TRKGEN

CSN

```
      SLA=SIN(TILAT(I)*RAD)
      CLO=COS(TILOG(I)*RAD)
      SLO=SIN(TILOG(I)*RAD)
      XG2=CLO*X1-SLA*SLO*Y1+CLA*SLO*Z1
      YG2=CLA*Y1+SLA*Z1
      ZG2=-SLO*X1-SLA*CLO*Y1+CLA*CLO*Z1
      CLO=COS(TLOG(I)*RAD)
      SLO=SIN(TLOG(I)*RAD)
      XG1=CLO
      ZG1=-SLO
      ARG=XG1*XG2+ZG1*ZG2
      IF(ARG-1.) 120,110,110
  110 ARG=1.
  120 IF(ARG+1.) 130,130,140
  130 ARG=-1.
  140 THD(I)=ACOS(ARG)/RAD
      IF(AINT) 150,160,160
  150 THD(I)=-THD(I)
  160 CONTINUE
      IF(THD(I)) 170,180,180
  170 THD(I)=360.+THD(I)
  180 CONTINUE
  200 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE SCOORD(NT,NS,ISHIP)
      COMMON/SECTOR/KSEC(64,5)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      ER=6378388.
      PR=6359911.
      PI = 3.141592654
      TOPI = PI*2.
      Q1=ER*ER
      RAD=.01745329252
      Q2=PR*PR
      J = ISHIP
      SLOP = SIN(SLOG(J)*RAD)
      CLOP = COS(SLOG(J)*RAD)
      SLAP = SIN(SLAT(J)*RAD)
      CLAP = COS(SLAT(J)*RAD)
      RHOS = SQRT(Q1*Q2/(Q1*SLAP*SLAP+Q2*CLAP*CLAP))
      RHOS=RHOS+SHT(J)
      DO 200 I=1,NS
      IF(I.EQ.J) GO TO 200
      SI=SIN(SLAT(I)*RAD)
      CO=COS(SLAT(I)*RAD)
      RHOT=SQRT(Q1*Q2/(Q1*SI*SI+Q2*CO*CO))
      RHOT=RHOT+SHT(I)
      CLA=COS(SLAT(I)*RAD)
      SLA=SIN(SLAT(I)*RAD)
      CLO=COS(SLOG(I)*RAD)
      SLO=SIN(SLOG(I)*RAD)
      X=RHOT*CLA*SLO
      Y=RHOT*SLA
      Z=RHOT*CLA*CLO
      XP = CLOP*X-SLOP*Z
      YP = -SLAP*SLOP*X+CLAP*Y-SLAP*CLOP*Z
      ZP = CLAP*SLOP*X+SLAP*Y+CLAP*CLOP*Z-RHOS
      DUM=XP*XP+YP*YP
      SDUM=SQRT(DUM)
      AYP=ABS(YP)
      K = I+NT
   40 AZ(K,J) = ATAN2(XP,YP)
   90 EL(K,J) = ATAN2(ZP,SDUM)
      RG(K,J) = SQRT(DUM+ZP*ZP)
      IF(AZ(K,J)) 150,160,160
  150 AZ(K,J) = TOPI+AZ(K,J)
  160 CONTINUE
      ISEC = AZ(K,J)*10.185916
      ISEC = ISEC+1
      KSEC(ISEC,ISHIP) = KSEC(ISEC,ISHIP) + 1
  200 CONTINUE
      RETURN
```

CSN

```
      SUBROUTINE TCOORD(NT,ISHIP)
      COMMON/SECTOR/KSEC(64,5)
      COMMON/PAR1/TLAT(20),TLOG(20),THT(20),THD(20)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      PI = 3.141592654
      TOPI = PI*2.
      ER=6378388.
      PR=6359911.
      RAD=.01745329252
      Q1=ER*ER
      Q2=PR*PR
      J = ISHIP
      SLOP = SIN(SLOG(J)*RAD)
      CLOP = COS(SLOG(J)*RAD)
      SLAP = SIN(SLAT(J)*RAD)
      CLAP = COS(SLAT(J)*RAD)
      RHOS = SQRT(Q1*Q2/(Q1*SLAP*SLAP+Q2*CLAP*CLAP))
      RHOS=RHOS+SHT(J)
      DO 200 I=1,NT
      SI=SIN(TLAT(I)*RAD)
      CO=COS(TLAT(I)*RAD)
      RHOT=SQRT(Q1*Q2/(Q1*SI*SI+Q2*CO*CO))
      RHOT=RHOT+THT(I)
      CLA=COS(TLAT(I)*RAD)
      SLA=SIN(TLAT(I)*RAD)
      CLO=COS(TLOG(I)*RAD)
      SLO=SIN(TLOG(I)*RAD)
      X=RHOT*CLA*SLO
      Y=RHOT*SLA
      Z=RHOT*CLA*CLO
      XP = CLOP*X-SLOP*Z
      YP = -SLAP*SLOP*X+CLAP*Y-SLAP*CLOP*Z
      ZP = CLAP*SLOP*X+SLAP*Y+CLAP*CLOP*Z-RHOS
      DUM=XP*XP+YP*YP
      SDUM=SQRT(DUM)
   40 AZ(I,J) = ATAN2(XP,YP)
   90 EL(I,J) = ATAN2(ZP,SDUM)
  100 CONTINUE
      RG(I,J) = SQRT(DUM+ZP*ZP)
      IF(AZ(I,J)) 150,160,160
  150 AZ(I,J) = AZ(I,J)+ TOPI
  160 CONTINUE
      ISEC = AZ(I,J)*10.185916
      ISEC = ISEC+1
      KSEC(ISEC,ISHIP) = KSEC(ISEC,ISHIP) + 1
  200 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DETFIL(IR,IS,ISEC,NT,NS)
C*********************************************************************
C SUBROUTINEDETFIL ASSIGNS DET. NO'S AND RANGE BINS TO TARGETS , IT IS
C THE FIRST STEP IN CREATING DETECTIONS FROM THE STABLIZED TARGET
C POSITIONS PROVIDED BY THE STIMULATION PROCESS.
C*********************************************************************
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      COMMON/DETFIL/IDET(3,5),IDTA(256,3,5)
      COMMON/LOAD/XSAV(3,20,3),AZINT(3,5),RVEL(3,5),RNGDIM(5,3)
      COMMON/TAG/ITAG(20,3,5,64)
      INTEGER TRATG
      ISECM1 = ISEC-1
      IF(ISECM1.EQ.0) ISECM1 = 64
      NTS = NT+NS
      J = IS
      DO 200 JRN=1,100
      LSTBIN(ISEC,JRN,IR,IS) =0
  200 CONTINUE
      AZLO = 5.625*(ISEC-1)
      AZHI = 5.625*(ISEC)
      DO 100 I=1,NTS
      K = I-NT

C IF TARGET IS ISHIP GO TO NEXT TARGET

      IF(K.EQ.J) GO TO 100

C IS TARGET IN SECTOR?

      AZR = AZ(I,J)*57.295779
      IF(AZR.LE.AZLO.OR.AZR.GT.AZHI) GO TO 100

C FIND RANGE BIN

      JRN = RG(I,J)/RNGDIM(IS,IR)
CCC   CHECK TO SEE IF TARGET IS WITHIN MINIMUM RANGE
      IF(JRN.EQ.0) GO TO 100

C CHECK TO SEE IF FILE IS FULL

      IF(IDET(IR,IS).GE.20)  GO TO 20
      IDET(IR,IS) = IDET(IR,IS) + 1
      GO TO 30
   20 CONTINUE
      IDET(IR,IS) = 1
   30 CONTINUE
      ID = IDET(IR,IS)
C LINK DETECTIONS TO TARGET.
      IDTA(ID,IR,IS) = I

CCC   CHECK TO SEE IF TARGET WAS DETECTED IN PREVIOUS SECTOR
      IF(ITAG(I,IR,IS,ISECM1).EQ.1) GO TO 199
      ITAG(I,IR,IS,ISEC) = 1
C LOAD DETECTION NUMBERS IN DETECTION FILE

      TRATG(ID) =0
      LNKBIN(ID,IR,IS) = LSTBIN(ISEC,JRN,IR,IS)
      LSTBIN(ISEC,JRN,IR,IS) = ID
  199 CONTINUE
  100 CONTINUE
      DO 150 IT=1,20
      ITAG(IT,IR,IS,ISECM1) = 0
  150 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE STAB1(NT,NS,ISHIP)
      COMMON/RADIAN/XX(20,20),YY(20,20),SS(20,20)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      COMMON/PART/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/NEW2/ AZD(20,20),ELD(20,20),OLDAZ(20,20)
      RAD=.01745329252
      PI = 3.141592654
      TOPI = PI*2.
      NTS=NT+NS
      J = ISHIP
      RR = ROLL(J)*RAD
      PP = PITCH(J)*RAD
      DO 200 I=1,NTS
      IF(I-NT.EQ.J) GO TO 200
      AZR = AZ(I,J)-(SHD(J)*RAD)
      ELR = EL(I,J)
      AAZS=ABS(AZR)
      IF(AAZS-PI  ) 210,240,240
  210 IF(AZR) 220,260,260
  220 AZR = TOPI+AZR
      GO TO 260
  240 IF(AZR) 250,260,260
  250 AZR=TOPI-AAZS
  260 CONTINUE
      C=COS(AZR)*SIN(PP)+TAN(ELR)*COS(PP)
      X=SIN(AZR)*COS(RR)+C*SIN(RR)
      Y=COS(AZR)*COS(PP)-TAN(ELR)*SIN(PP)
      XX(I,J) = X
      YY(I,J) = Y
   40 AZD(I,J) = ATAN2(X,Y)/RAD
   50 IF(AZD(I,J)) 60,65,65
   60 AZD(I,J)= 360.+ AZD(I,J)
   65 CONTINUE
      C=COS(ELR)*COS(AZR)*SIN(PP)+SIN(ELR)*COS(PP)
      S=C*COS(RR)-COS(ELR)*SIN(AZR)*SIN(RR)
      SS(I,J) = S
  100 ELD(I,J) = ARSIN(S)/RAD
  110 CONTINUE
  150 CONTINUE
  200 CONTINUE
      RETURN
      END
```

```
CSN


0001          SUBROUTINE NOISY(IR,IS,ISEC)
        C**********************************************************************
        C SUBROUTINE NOISY OUTPUTS THE NOISY X,Y,Z STABLIZED COORDINATES
        C OF EVERY DECTION IN SECTOR ISEC
        C**********************************************************************
0002          COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
             1LSTBIN(64,100,3,5),XS(20),THRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
0003          COMMON/DETFIL/IDET(3,5),IDTA(256,3,5)
0004          COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
0005          COMMON/TRUE/XYZTRU(20,3,3,5)
        C GO THRU ALL RANGE BINS IN SECTOR
0006          DO 100 JRN=1,100
        C FIRST DETECTION IN RANGE BIN
0007          ID = LSTBIN(ISEC,JRN,IR,IS)
0008       10 CONTINUE
        C CHECK TO SEE IF THERE ARE ANY OTHER DETECTIONS IN RANGE BINS
0009          IF(ID.EQ.0) GO TO 100
        C GET TARGET NUMBER CORRESPONDING TO DETECTION
0010          IT = IDTA(ID,IR,IS)
0011          XYZTRU(ID,1,IR,IS) = RG(IT,IS)*SIN(AZ(IT,IS))*COS(EL(IT,IS))
0012          XYZTRU(ID,2,IR,IS) = RG(IT,IS)*COS(AZ(IT,IS))*COS(EL(IT,IS))
0013          XYZTRU(ID,3,IR,IS) = RG(IT,IS)*SIN(EL(IT,IS))
        C GET NOISY STABILIZED RANGE,AZ,EL.
0014          CALL STAB2(IT,IS,IR)
0015          XYZMS(ID,1,IR,IS) = RG(IT,IS)*SIN(AZ(IT,IS)) * COS(EL(IT,IS))
0016          XYZMS(ID,2,IR,IS) = RG(IT,IS)*COS(AZ(IT,IS)) * COS(EL(IT,IS))
0017          XYZMS(ID,3,IR,IS) = RG(IT,IS)*SIN(EL(IT,IS))
0018          TMS(ID,IR,IS) = THRK(ISEC,IR,IS)
0019          ID = LNKBIN(ID,IR,IS)
0020          GO TO 10
0021      100 CONTINUE
0022          RETURN
0023          END
```

CSN

```
      SUBROUTINE STAB2(I,J,K)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      COMMON/RANDUM/IRAN(20,5)
      COMMON/NDECK/AZND(20,5,3),ELND(20,5,3),RNND(20,5,3)
      COMMON/RADIAN/XX(20,20),YY(20,20),SS(20,20)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/COV1/ SIGAZD(20,2),SIGELD(20,2),RHOD(20,2)
      COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/NEW2/ AZD(20,20),ELD(20,20),OLDAZ(20,20)
      DIMENSION RAN(100)
      RAD=.01745329252
      PI = 3.141592654
      CALL SETVR(IRAN(I,J))
      NNN = 6
      CALL VRANF(RAN,NNN)
      ARG = -2.*ALOG(RAN(1))
      RND = SQRT(ARG)* COS(6.2832*RAN(2))
      AZR = ATAN2(XX(I,J),YY(I,J)) +SIGAZD(J,K)*RND*RAD
      ARG = -2.*ALOG(RAN(3))
      RND = SQRT(ARG)* COS(6.2832*RAN(4))
      ELR = ARSIN(SS(I,J)) +SIGELD(J,K)*RND*RAD
      ARG = -2.*ALOG(RAN(5))
      RND = SQRT(ARG)* COS(6.2832*RAN(6))
      IRANN = RAN(6)*10000.
      IRANN = (IRANN*2)+1
      IF(IRANN.EQ.IRAN(I,J)) IRANN=IRANN+1
      IRAN(I,J) = IRANN
      RG(I,J) = RG(I,J) + RHOD(J,K)*RND
      AZND(I,J,K) = AZR
      ELND(I,J,K) = ELR
      RNND(I,J,K) = RG(I,J)
      RR = ROLL(J)*RAD
      PP = PITCH(J)*RAD
      X=-SIN(RR)*SIN(ELR)+COS(RR)*SIN(AZR)*COS(ELR)
      C=COS(RR)*SIN(ELR)+SIN(RR)*SIN(AZR)*COS(ELR)
      Y=COS(PP)*COS(AZR)*COS(ELR)+SIN(PP)*C
   40 AZE=ATAN2(X,Y)
      AZ(I,J) = AZE +SHD(J)*RAD
      C=COS(RR)*SIN(ELR)+SIN(RR)*SIN(AZR)*COS(ELR)
      S=-SIN(PP)*COS(AZR)*COS(ELR)+COS(PP)*C
  100 ELE=ARSIN(S)
      EL(I,J) = ELE
      RETURN
      END
```

```
CSN

0001          SUBROUTINE PREDIC(ISEC,IR,IS)
        C SUBROUTINE PREDIC DETERMINES THE POSITION OF TRACKS AT THE SECTOR
        C CROSSING TIME. BASED ON THIS PREDICTED POSITION IT ALSO MAKES
        C ADJUSTMENTS TO THE SECTOR TRACK FILES.
0002          COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
             1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
0003          COMMON/DUMSEC/DUMSX(64,5),DUMID(512,5)
0004          COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
             10,20),LNKID(20),TIMLNK(20,5),RBLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
             2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
0005          COMMON/PREDIC/RAEDUM(256,3,5),XYZDUM(256,3,5)
0006          DIMENSION X(9),TDUM(256,5)
0007          INTEGER DUMSX,DUMID,PTFST,TRKST
0008          PI = 3.141592654
0009          TOPI = PI*2.
0010          NT = DUMSX(ISEC,IS)
0011          TIME = TMRK(ISEC,IR,IS)
0012       10 CONTINUE
0013          IF(NT.EQ. 0) GO TO 999
0014          R = 0.
0015          MT = TRKST(NT,IS)
0016          KS = PTFST(NT,IS)
0017          IF(KS.EQ.IS) GO TO 20
0018          XT = XSMO(1,MT,KS)
0019          YT = XSMO(3,MT,KS)
0020          ZT = XSMO(5,MT,KS)
0021          CALL TRANSF(XT,YT,ZT,X(1),X(2),X(3),IS,KS)
0022          XT = XSMO(2,MT,KS)
0023          YT = XSMO(4,MT,KS)
0024          ZT = XSMO(6,MT,KS)
0025          CALL VTRANS(XT,YT,ZT,X(4),X(5),X(6),IS,KS)
0026          GO TO 30
0027       20 CONTINUE
0028          X(1) = XSMO(1,MT,IS)
0029          X(2) = XSMO(3,MT,IS)
0030          X(3) = XSMO(5,MT,IS)
0031          X(4) = XSMO(2,MT,IS)
0032          X(5) = XSMO(4,MT,IS)
0033          X(6) = XSMO(6,MT,IS)
0034       30 CONTINUE
0035          DO 35 I=1,3
0036          J=I+3
0037          XYZDUM(NT,I,IS) = X(I)+X(J)*(TIME-TIMLNK(MT,KS))
0038          R = R+ XYZDUM(NT,I,IS)*XYZDUM(NT,I,IS)
0039       35 CONTINUE
0040          NTT = DUMID(NT,IS)
0041          TDUM(NT,IS) = TIME
0042          RAEDUM(NT,1,IS) = SQRT(R)
0043          RAEDUM(NT,2,IS) = ATAN2(XYZDUM(NT,1,IS),XYZDUM(NT,2,IS))
```

PREDIC

ISN

```
      TEM = SQRT(XYZDUM(NT,1,IS)*XYZDUM(NT,1,IS) + XYZDUM(NT,2,IS)*XYZDU
     1M(NT,2,IS))
      RAEDUM(NT,3,IS) = ATAN2(XYZDUM(NT,3,IS),TEM)
      IF(RAEDUM(NT,2,IS).GE. 0.) GO TO 50
      RAEDUM(NT,2,IS) = RAEDUM(NT,2,IS) + TOPI
      GO TO 60
   50 CONTINUE
      IF(RAEDUM(NT,2,IS).LE.TOPI) GO TO 60
      RAEDUM(NT,2,IS) = RAEDUM(NT,2,IS)- TOPI
   60 CONTINUE
      IF(ISEC.EQ.1 .AND. RAEDUM(NT,2,IS).GT.PI) GO TO 666
      IF(ISEC.EQ.64 .AND. RAEDUM(NT,2,IS).LT.PI) GO TO 777
      AZSEC2= ISEC*(TOPI/64.)
      AZSEC1=(ISEC-1)*(TOPI/64.)
      IF(RAEDUM(NT,2,IS).GE.AZSEC1) GO TO 70
      ISM = ISEC-1
  665 CONTINUE
      AZSECM = (ISM-1)*(TOPI/64)
      IF(RAEDUM(NT,2,IS).GE.AZSECM) GO TO 667
      ISM = ISM-1
      GO TO 665
  667 CONTINUE
      CALL DUMDRP(NT,ISEC,IS)
      CALL DUMNEW(NT,ISM,IS)
      GO TO 90
   70 CONTINUE
      IF(RAEDUM(NT,2,IS).LE.AZSEC2) GO TO 90
      ISP =ISEC+1
   65 CONTINUE
      AZSECP = ISP*(TOPI/64.)
      IF(RAEDUM(NT,2,IS).LE.AZSECP) GO TO 778
      ISP = ISP+1
      GO TO 65
  778 CONTINUE
      CALL DUMDRP(NT,ISEC,IS)
      CALL DUMNEW(NT,ISP,IS)
   90 CONTINUE
      NT = NTT
      GO TO 10
  666 CONTINUE
      ISM = 64
      GO TO 667
  777 CONTINUE
      ISP = 1
      GO TO 778
  999 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CORRAS(ISHIP,ISEC,IRAD)
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/NDECK/AZND(20,5,3),ELND(20,5,3),RNND(20,5,3)
      COMMON/DUMSEC/DUMSX(64,5),DUMID(512,5)
      COMMON/PREDIC/RAEDUM(256,3,5),XYZDUM(256,3,5)
      COMMON/CORAS/DETSX(256),ISTOR(256),DETID(512)
      COMMON/LOAD/XSAV(3,20,3),AZINT(3,5),RVEL(3,5),RNGDIM(5,3)
      COMMON/DIST/SD(256)
      INTEGER DUMID,DUMSX,DETSX,DETID,TRATG
      DIMENSION IRADET(10),SDIST(10)
C     PICK TRACKS OUT OF DUMMY SECTOR FILE
      NT = DUMSX(ISEC,ISHIP)
    5 CONTINUE
C     HAVE ALL THE TRACKS IN THIS SECTOR BEEN CONSIDERED
      IF(NT.EQ. 0) GO TO  999
  200 CONTINUE
      LOC = DETSX(NT)
      IF(LOC.EQ.0) GO TO 300
      DETSX(NT) = DETID(LOC)
      DETID(LOC) = 0
      IDRP = 0
      CALL NEWLOC(LOC,IDRP)
      GO TO 200
  300 CONTINUE
      DO 8 I=1,10
      SDIST(I) = 0
    8 CONTINUE
C     FIND RANGE BIN
      JRNG = RAEDUM(NT,1,ISHIP)/RNGDIM(ISHIP,IRAD)
C     START PICKING DETECTIONS OUT OF RANGE BINS
      K = 1
      DO 40 II=1,3
      DO 40 JJ=1,3
      J =JRNG-2+JJ
      I = ISEC-2+II
      IF(I.EQ.0) I=64
      IF(J.LE.0) GO TO 40
      IDETNO = LSTBIN(I,J,IRAD,ISHIP)
   10 CONTINUE
C     ARE THERE ANY DETECTIONS LEFT IN BIN I,J
      IF(IDETNO.EQ. 0) GO TO 40
CCC   IF PREDIC HAS MOVED TRACK INTO NEW SECTOR IT MAY HAVE BEEN PREVIOUSLY
CCC   ASSIOIATED WITH THIS DETECTION
      IF(TRATG(IDETNO).EQ.NT) GO TO 40
C     ORDER DETECTIONS ACCORDING TO STATISTICAL DISTANCE
      NN = K
      CALL COVOWN(IRAD,ISHIP,IDETNO,NT,SD1,ISEC)
   20 CONTINUE
      IF(NN.EQ.1) GO TO 30
```

CORRAS

```
         IF(SD1.GT.SDIST(NN-1)) GO TO 30
         SDIST(NN) = SDIST(NN-1)
         IRADET(NN) = IRADET(NN-1)
         NN= NN-1
         GO TO 20
      30 SDIST(NN) = SD1
         IRADET(NN) = IDETNO
         IDETNO = LNKBIN(IDETNO,IRAD,ISHIP)
         K = K+1
         GO TO 10
      40 CONTINUE
C        WERE THERE ANY DETECTIONS IN CONT. BINS
         IF(K.NE.1) GO TO 60
      50 CONTINUE
         NT = DUMID(NT,ISHIP)
         GO TO 5
      60 CONTINUE
         J = K-1
      70 CONTINUE
         IF(J.EQ.0) GO TO 80
C        PLACE DETECTIONS IN LINKED FILE
C        SMALLEST SD IS LAST ONE IN
         IDRP = 1
         CALL NEWLOC(LOC,IDRP)
         ISTOR(LOC) = IRADET(J)
         DETID(LOC) = DETSX(NT)
         DETSX(NT)  = LOC
         SD(LOC) = SDIST(J)
         J = J-1
         GO TO 70
      80 CONTINUE
C        ASSOCIATION PROCESS
         NTA = NT
      90 CONTINUE
         LOC = DETSX(NTA)
         IDETNO = ISTOR(LOC)
C        ARE ANY OTHER TRACKS ASSOCIATED WITH THIS DETECTION
         IF(TRATG(IDETNO).NE. 0) GO TO 100
         TRATG(IDETNO) = NTA
         GO TO 50
     100 CONTINUE
         NTT = TRATG(IDETNO)
         LOCC = DETSX(NTT)
         IF(SD(LOC).GE.SD(LOCC)) GO TO 110
         TRATG(IDETNO) = NTA
         NTA = NTT
     110 CONTINUE
         LOC = DETSX(NTA)
         DETSX(NTA) = DETID(LOC)
         IF(DETSX(NTA).EQ.0) GO TO 50
         IDRP = 0
         CALL NEWLOC(LOC,IDRP)
         GO TO 90
     999 CONTINUE
         RETURN
         END
```

CSN

```
      SUBROUTINE COVPHN(IRAD,ISHIP,IDETNO,NT,SD,ISEC)
      COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/KAL2/ PS(20,20),PP(20,20),COVS(20,20),COVM(20,20),XP(20)
      COMMON/KAL4/IFIRST(20,5),DIM1,DIM2,DIM3
      COMMON/DETFIL/IDET(3,5),IDTA(256,3,5)
      COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
     10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
     2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
      COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
      COMMON/STATIC/N(9),N2(3)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      COMMON/NDECK/AZND(20,5,3),ELND(20,5,3),RNND(20,5,3)
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/LODE/NUMTAR
      DIMENSION H(3,3),P(3,3),HP(3,3),AJS(3,3)
      DIMENSION A2(3,3),NN2T(9,9),SUM2(3,3)
      REAL N,N2,NNT,NN2T
      INTEGER TRKST,DIM1,DIM2,DIM3,PTFST,DUMST
      RAD = .01745329252
      MT = TRKST(NT,ISHIP)
      KSHIP = PTFST(NT,ISHIP)
      IF(KSHIP.NE.ISHIP) GO TO 85
      JJ = ISHIP
      DO 120  I=1,3
      DO 120  J=1,3
      H(I,J) = 0.
  120 CONTINUE
      H(1,1) = COS(SHD(JJ)*RAD)
      H(1,2) = SIN(SHD(JJ)*RAD)
      H(2,1) =-SIN(SHD(JJ)*RAD)
      H(2,2) = H(1,1)
      H(3,3) = 1.
      ROLRAD = ROLL(JJ)*RAD
      PITRAD = PITCH(JJ)*RAD
      P(1,1) = COS(ROLRAD)
      P(1,2) = 0.
      P(1,3) = -SIN(ROLRAD)
      P(2,1) = SIN(ROLRAD)* SIN(PITRAD)
      P(2,2) = COS(PITRAD)
      P(2,3) = COS(ROLRAD)* SIN(PITRAD)
      P(3,1) = SIN(ROLRAD)* COS(PITRAD)
      P(3,2) = -SIN(PITRAD)
      P(3,3) = COS(ROLRAD)* COS(PITRAD)
      DO 130  I=1,3
      DO 130  J=1,3
      HP(I,J)= 0.
      DO 130  IJ=1,3
      HP(I,J) = HP(I,J)+ H(I,IJ)* P(IJ,J)
  130 CONTINUE
      IF(IDETNO.NE.0) GO TO 144
C CALL FROM LOAD
      IT = NUMTAR
      GO TO 148
  144 CONTINUE
      IT = IDTA(IDETNO,IRAD,ISHIP)
  148 CONTINUE
      EL2 = ELND(IT,ISHIP,IRAD)
      AZ2 = AZND(IT,ISHIP,IRAD)
      RN2 = RNND(IT,ISHIP,IRAD)
      A2(1,1)= COS(EL2)*SIN(AZ2)
      A2(1,2)= -RN2*SIN(EL2)*SIN(AZ2)
      A2(1,3) = RN2*COS(EL2)*COS(AZ2)
      A2(2,1) =COS(EL2)*COS(AZ2)
```

COVOWN

```
      A2(2,2)= -RN2*SIN(EL2)*COS(AZ2)
      A2(2,3)= -RN2*COS(EL2)*SIN(AZ2)
      A2(3,1)= SIN(EL2)
      A2(3,2)= RN2*COS(EL2)
      A2(3,3)=0.
      DO 150 I=1,3
      DO 150 J=1,3
      AJS(I,J) = 0.
      DO 150 IJ =1,3
      AJS(I,J) = AJS(I,J) + HP(I,IJ)*A2(IJ,J)
  150 CONTINUE
      DO 30 I=1,3
      DO 30 K=1,3
      NN2T(I,K)=0.
   30 CONTINUE
      DO 40 I=1,3
      NN2T(I,I)=N2(I)*N2(I)
   40 CONTINUE
      DO 70 I=1,3
      DO 70 J=1,3
      SUM2(I,J)=0.
      DO 70 IJ=1,3
      SUM2(I,J)=SUM2(I,J)+AJS(I,IJ)*NN2T(IJ,J)
   70 CONTINUE
      DO 80 I=1,3
      DO 80 J=1,3
      COVM(I,J) = 0.
      DO 80 IJ=1,3
      COVM(I,J) = COVM(I,J) +SUM2(I,IJ)*AJS(J,IJ)
   80 CONTINUE
C CALL FROM LOAD
      IF(IDETNO.EQ.0) GO TO 100
      DO 85 I=1,3
      DO 85 J=1,3
      COVMS(IDETNO,I,J,IRAD,ISHIP) = COVM(I,J)
   85 CONTINUE
      ZM1 = XYZMS(IDETNO,1,IRAD,ISHIP)
      ZM2 = XYZMS(IDETNO,2,IRAD,ISHIP)
      ZM3 = XYZMS(IDETNO,3,IRAD,ISHIP)
      KFLAG = 2
      IF(KSHIP.EQ.ISHIP) GO TO 88
      X = ZM1
      Y = ZM2
      Z = ZM3
      CALL TRANSF(X,Y,Z,ZM1,ZM2,ZM3,KSHIP,ISHIP)
      CALL COVLNK(ISHIP,IDETNO,IRAD,KSHIP)
   88 CONTINUE
      NNT = DUMST(KSHIP,MT,KSHIP)
      TDEL = TMS(IDETNO,IRAD,ISHIP) -TIMLNK(MT,KSHIP)
      DO 66 I=1,6
      XS(I) = YSMO(I,MT,KSHIP)
      DO 66 J=1,6
      PS(I,J) = COVSMO(I,J,MT,KSHIP)
   66 CONTINUE
      IF(IFIRST(MT,KSHIP).NE.0) GO TO 90
      XP(1) = XSMO(1,MT,KSHIP)
      XP(3) = XSMO(3,MT,KSHIP)
      XP(5) = XSMO(5,MT,KSHIP)
   90 CONTINUE
      CALL KALMAN(TDEL,MT,ZM1,ZM2,ZM3,KSHIP,KFLAG,SD)
  100 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE COVLNK(KS,ID,IR,JSHIP)
      COMMON/NDECK/AZND(20,5,3),ELND(20,5,3),RNND(20,5,3)
      COMMON/DETFIL/IDET(3,5),IDTA(256,3,5)
      COMMON/STATIC/N(9),N2(3)
      COMMON/KAL2/ PS(20,20),PP(20,20),COVS(20,20),COVM(20,20),XP(20)
      COMMON/NEW2/ AZD(20,20),ELD(20,20),OLDAZ(20,20)
      COMMON/UPDATE/ILOC(20),TIMLAG,TESTR(512,3),TLAST(256,5)
      COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
      COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
      DIMENSION H(3,3),P(3,3),HP(3,3),THP(3,3),AJS(3,3)
      DIMENSION A(3,9),A2(3,3),NNT(9,9),NN2T(9,9),T(3,3)
      DIMENSION SUM(3,9),SUM2(3,3),ANNA1(3,3),ANNA2(3,3),COV(3,3)
      REAL LA1,LA2,LA
      REAL N,N2,NNT,NN2T
      RAD = .01745329252
      JJ = KS
      IT = IDTA(ID,IR,KS)
      EL = ELND(IT,KS,IR)
      AZ = AZND(IT,KS,IR)
      RN = RNND(IT,KS,IR)
      DO 120  I=1,3
      DO 120  J=1,3
      H(I,J) = 0.
  120 CONTINUE
      H(1,1) = COS(SHD(JJ)*RAD)
      H(1,2) = SIN(SHD(JJ)*RAD)
      H(2,1) =-SIN(SHD(JJ)*RAD)
      H(2,2) = H(1,1)
      H(3,3) = 1.
      ROLRAD = ROLL(JJ)*RAD
      PITRAD = PITCH(JJ)*RAD
      P(1,1) = COS(ROLRAD)
      P(1,2) = 0.
      P(1,3) = -SIN(ROLRAD)
      P(2,1) = SIN(ROLRAD)* SIN(PITRAD)
      P(2,2) = COS(PITRAD)
      P(2,3) = COS(ROLRAD)* SIN(PITRAD)
      P(3,1) = SIN(ROLRAD)* COS(PITRAD)
      P(3,2) = -SIN(PITRAD)
      P(3,3) = COS(ROLRAD)* COS(PITRAD)
      DO 130  I=1,3
      DO 130  J=1,3
      HP(I,J)= 0.
      DO 130  IJ=1,3
      HP(I,J) = HP(I,J)+ H(I,IJ)* P(IJ,J)
  130 CONTINUE
      TH2 = SLAT(JSHIP)*RAD
      LA = (SLOG(JJ)*RAD)- (SLOG(JSHIP)*RAD)
      TH1 = SLAT(JJ)*RAD
      T(1,1) = COS(LA)
      T(1,2) = SIN(TH1)* SIN(LA)
      T(1,3) = COS(TH1)* SIN(LA)
      T(2,1) = SIN(TH2)* SIN(LA)
      T(2,2) = SIN(TH1)* SIN(TH2)*COS(LA) +COS(TH1)*COS(TH2)
      T(2,3) =-COS(TH1)* SIN(TH2)*COS(LA) +SIN(TH1)*COS(TH2)
      T(3,1) =-COS(TH2)* SIN(LA)
      T(3,2) = COS(TH1)* SIN(TH2) -SIN(TH1)* COS(TH2)*COS(LA)
      T(3,3) = COS(TH1)* COS(TH2)*COS(LA) +SIN(TH1)*SIN(TH2)
      DO 140 I=1,3
      DO 140 J=1,3
```

COVLNK

CSN

```
      THP(I,J)= 0.
      DO 140 IJ=1,3
      THP(I,J) = THP(I,J)+ T(I,IJ)*HP(IJ,J)
  140 CONTINUE
      A(1,1) = T(1,1)
      A(1,2) = T(1,2)
      A(1,3) = T(1,3)
      A(1,4) = THP(1,1)*COS(EL)*SIN(AZ) +THP(1,2)*COS(EL)*COS(AZ)+THP(1,
     13)*SIN(EL)
      A(1,5) = -THP(1,1)*RN* SIN(EL)*SIN(AZ) - THP(1,2)*RN*SIN(EL)*
     2COS(AZ) +THP(1,3)*RN*COS(EL)
      A(1,6) = THP(1,1)*RN*COS(EL)*COS(AZ) - THP(1,2)*RN*COS(EL)*SIN(AZ)
      A(1,7) =-1
      A(1,8) = 0
      A(1,9) = 0
      A(2,1) = T(2,1)
      A(2,2) = T(2,2)
      A(2,3) = T(2,3)
      A(2,4) = THP(2,1)*COS(EL)*SIN(AZ) +THP(2,2)*COS(EL)*COS(AZ)+
     3THP(2,3)*SIN(EL)
      A(2,5) = -THP(2,1)*RN*SIN(EL)*SIN(AZ) - THP(2,2)*RN*SIN(EL)*COS(AZ
     4) + THP(2,3)*RN*COS(EL)
      A(2,6) = THP(2,1)*RN*COS(EL)*COS(AZ) - THP(2,2)*RN*COS(EL)*SIN(AZ)
      A(2,7) = 0
      A(2,8) = -1
      A(2,9) = 0
      A(3,1) = T(3,1)
      A(3,2) = T(3,2)
      A(3,3) = T(3,3)
      A(3,4) = THP(3,1)*COS(EL)*SIN(AZ) + T(3,2)*COS(EL)*COS(AZ)+ THP(3,
     53)* SIN(EL)
      A(3,5) = -THP(3,1)*RN*SIN(EL)*SIN(AZ) -THP(3,2)*RN*SIN(EL)*COS(AZ)
     6+ THP(3,3)*RN*COS(EL)
      A(3,6) = -THP(3,1)*RN*COS(EL)*COS(AZ)-THP(3,2) *RN*COS(EL)*SIN(AZ)
      A(3,7) = 0
      A(3,8) = 0
      A(3,9) =-1
      DO 10 I=1,9
      DO 10 K=1,9
      NNT(I,K)=0.
10    CONTINUE
      DO 20 I=1,9
      NNT(I,I)=N(I)*N(I)
20    CONTINUE
      DO 50 I=1,3
      DO 50 J=1,9
      SUM(I,J)=0.
      DO 50 IJ=1,9
      SUM(I,J)=SUM(I,J)+A(I,IJ)*NNT(IJ,J)
50    CONTINUE
 90   CONTINUE
      DO 60 I=1,3
      DO 60 J=1,3
      COVM(I,J) = 0.
      DO 60 IJ=1,9
      COVM(I,J)= COVM(I,J) + SUM(I,IJ)*A(J,IJ)
60    CONTINUE
 45   FORMAT(5X,9F14.3,//)
      RETURN
      END
```

```
      SUBROUTINE KALMAN(TDEL,ITAR,ZM1,ZM2,ZM3,ISHIP,KFLAG,SD)
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/KAL1/ PHI(20,20),G(20,20),H(20,20)
      COMMON/KAL2/ PS(20,20),PP(20,20),COVS(20,20),COVM(20,20),XP(20)
      COMMON/KAL4/IFIRST(20,5),DIM1,DIM2,DIM3
      COMMON/TRIANG/ COVI(20,20),UP(20,20),FL(20,20)
      COMMON/INVERT/R(20,20),RI(20,20)
      DIMENSION XPMU(6),SUM(6),SUM2(3),XMMU(3)
      DIMENSION ZM(20),HT(20,20)
      DIMENSION DM1(20,20),DM2(20,20),DM3(20,20)
      DIMENSION VACI(3,3),D3M(3,3),D2M(3,3)
      DIMENSION D4M(6,6),D5M(6,6),VOC2(6,6)
      INTEGER DIM1,DIM2,DIM3
      ZM(1) = ZM1
      ZM(2) = ZM2
      ZM(3) = ZM3
      M = DIM1
      N = DIM2
      NS = DIM3
      JFIRST = IFIRST(ITAR,ISHIP)
      IF(JFIRST.LT. 2 .AND. KFLAG .EQ. 2) GO TO  1299
      IF(JFIRST) 10,165,100
   10 CONTINUE
C     FIRST TIME THRU
      DO 120 I=1,N
      DO 120 J=1,N
C     SMOOTHED COVARIANCE MATRIX
      PS(I,J) = 0
  120 CONTINUE
      PS(1,1) = COVM(1,1)
      PS(3,3) = COVM(2,2)
      PS(5,5) = COVM(3,3)
      PS(1,3) = COVM(1,2)
      PS(3,1) = PS(1,3)
      PS(1,5) = COVM(1,3)
      PS(5,1) = PS(1,5)
      PS(3,5) = COVM(2,3)
      PS(5,3) = PS(3,5)
C     PREDICTED POSITION VECTOR
      DO 160 I=1,N
      XP(I) = XS(I)
  160 CONTINUE
      GO TO 1200
  165 CONTINUE
C     SECOND TIME THRU
      PS(2,2) =(COVM(1,1)+ PS(1,1))/(TDEL*TDEL)
      PS(2,4) = (COVM(1,2)+ PS(1,3))/(TDEL*TDEL)
      PS(4,4) = (COVM(2,2)+ PS(3,3))/(TDEL*TDEL)
      PS(2,6) = (COVM(1,3)+ PS(1,5))/(TDEL*TDEL)
```

KALMAN

CSN

```
      PS(4,2) = PS(2,4)
      PS(6,2) = PS(2,6)
      PS(1,1) = COVM(1,1)
      PS(1,2) = COVM(1,1)/TDEL
      PS(2,1) = PS(1,2)
      PS(1,3) = COVM(1,2)
      PS(3,1) = PS(1,3)
      PS(4,6) =(COVM(2,3)+ PS(3,5))/(TDEL*TDEL)
      PS(6,4) = PS(4,6)
      PS(6,6) = (COVM(3,3)+PS(5,5))/(TDEL*TDEL)
      PS(1,4) = COVM(1,2)/TDEL
      PS(4,1) = PS(1,4)
      PS(2,3) = COVM(1,2)/TDEL
      PS(3,2) = PS(2,3)
      PS(3,3) = COVM(2,2)
      PS(3,4) = COVM(2,2)/TDEL
      PS(4,3) = PS(3,4)
      PS(1,5) = COVM(1,3)
      PS(5,1) = PS(1,5)
      PS(2,5) = COVM(1,3)/TDEL
      PS(5,2) = PS(2,5)
      PS(3,5) = COVM(2,3)
      PS(5,3) = PS(3,5)
      PS(4,5) = COVM(2,3)/TDEL
      PS(5,4) = PS(4,5)
      PS(5,5) = COVM(3,3)
      PS(1,6) = COVM(1,3)/TDEL
      PS(6,1) = PS(1,6)
      PS(3,6) = COVM(2,3)/TDEL
      PS(6,3) = PS(3,6)
      PS(5,6) = COVM(3,3)/TDEL
      PS(6,5) = PS(5,6)
      DO 155 I=1,N
      DO 155 J=1,N
C     PREDICTED COVARIANCE MATRIX
      PP(I,J) = PS(I,J)
  155 CONTINUE
      DO 170 I=1,N
      DO 170 J=1,M
      DM1(I,J)=0.
      DO 170 K=1,N
      DM1(I,J) = DM1(I,J) + PS(I,K)*H(J,K)
  170 CONTINUE
      DO 180 I=1,M
      DO 180 J=1,M
      COVI(I,J) = COVM(I,J)
  180 CONTINUE
      CALL UPPERT(M)
      DO 190 I=1,M
      DO 190 J=1,M
```

AD-A079 951 NAVAL RESEARCH LAB WASHINGTON DC F/G 9/2

MULTIPLE PLATFORM SENSOR INTEGRATION MODEL: MULSIM COMPUTER PRO--ETC(U)

DEC 79 A GRINDLAY

UNCLASSIFIED NRL-8358 SBIE-AD-E000 349 NL

2 of 2

END

DATE FILMED

2 80

DDC

KALMAN

CSN

```
      R(I,J) = UP(J,I)
  190 CONTINUE
      CALL INVERL(M)
      DO 195 I=1,M
      DO 195 J=1,M
      DM2(I,J) = 0.
      DO 195 K=1,M
      DM2(I,J) = DM2(I,J) +RI(I,K)*RI(J,K)
  195 CONTINUE
      DO 215 I=1,N
      DO 215 J=1,M
      WT(I,J) = 0.
      DO 215 K=1,M
      WT(I,J) = WT(I,J) +DM1(I,K)*DM2(K,J)
  215 CONTINUE
      GO TO 890
  100 CONTINUE
C     SET UP TRANSITION MATRIX
      DO 105  I=1,N
      DO 105  J=1,N
      PHI(I,J) = 0.
      PHI(I,I) = 1.
  105 CONTINUE
      PHI(1,2) = TDEL
      PHI(3,4) = TDEL
      PHI(5,6) = TDEL
C
C     COMPUTE PREDICTION COVARIANCE
C
      DO 150 I=1,N
      DO 150 J=1,N
      DM1(I,J)=0.
      DO 150 K=1,N
  150 DM1(I,J)=DM1(I,J)+PS(I,K)*PHI(J,K)
      DO 210 I=1,N
      DO 210 J=1,N
      DM2(I,J)=0.
      DO 200 K=1,N
  200 DM2(I,J)=DM2(I,J)+PHI(I,K)*DM1(K,J)
  210 PR(I,J)=DM2(I,J)
      IF(NS) 320,320,240
  240 CONTINUE
      DO 250 I=1,NS
      DO 250 J=1,N
      DM1(I,J)=0.
      DO 250 K=1,NS
  250 DM1(I,J)=DM1(I,J)+COVS(I,K)*G(J,K)
      DO 310 I=1,N
      DO 310 J=1,N
      DM2(I,J)=0.
```

KALMAN

CSN

```
      DO 300 K=1,NS
  300 DM2(I,J)=DM2(I,J)+G(I,K)*DM1(K,J)
  310 PP(I,J)=PP(I,J)+DM2(I,J)
  320 CONTINUE
      DO 315 I=1,N
      DO 315 J=1,N
  315 PP(I,J) = PP(I,J)*EXP(.05*TDEL)
C
C     COMPUTE FILTER WEIGHTS
C
      DO 350 I=1,N
      DO 350 J=1,M
      DM1(I,J)=0.
      DO 350 K=1,N
  350 DM1(I,J)=DM1(I,J)+PP(I,K)*H(J,K)
      DO 410 I=1,M
      DO 410 J=1,M
      DM2(I,J)=0.
      DO 400 K=1,N
  400 DM2(I,J)=DM2(I,J)+H(I,K)*DM1(K,J)
  410 VACI(I,J) = DM2(I,J)+COVM(I,J)
      CALL MAT(VACI,M,M,D3M,M,M,D2M,M,M,XXX,4)
      DO 550 I=1,N
      DO 550 J=1,M
      WT(I,J)=0.
      DO 550 K=1,M
  550 WT(I,J) = WT(I,J)+DM1(I,K)*D2M(K,J)
C
C     UPDATE SMOOTHED COVARIANCE
C
      DO 600 I=1,N
      DO 600 J=1,N
      DM1(I,J)=0.
      DO 600 K=1,M
  600 DM1(I,J)=DM1(I,J)+WT(I,K)*H(K,J)
      DO 660 I=1,N
      DO 660 J=1,N
      DM2(I,J)=0.
      DO 650 K=1,N
  650 DM2(I,J)=DM2(I,J)+DM1(I,K)*PP(K,J)
  660 PS(I,J)=PP(I,J)-DM2(I,J)
C
C     FILTER UPDATE
C
      DO 900 I=1,N
      XP(I)=0.
      DO 900 J=1,N
  900 XP(I)=XP(I)+PHI(I,J)*XS(J)
  890 CONTINUE
      DO 960 I=1,M
```

KALMAN

CSN

```
      DM1(I,1)=0.
      DO 950 J=1,N
  950 DM1(I,1)=DM1(I,1)+H(I,J)*XP(J)
  960 DM2(I,1)=ZM(I)-DM1(I,1)
      DO 1010 I=1,N
      DM1(I,1)=0.
      DO 1000 J=1,M
 1000 DM1(I,1)=DM1(I,1)+HT(I,J)*DM2(J,1)
 1010 XS(I)=XP(I)+DM1(I,1)
 1200 CONTINUE
      IF(KFLAG.NE.2) GO TO 1300
C
C     CALCULATE STATISTICAL DISTANCE
C
      DO 166 I=1,N
      XPMU(I) = -DM1(I,1)
      DO 166 J=1,N
      VOC2(I,J) = PP(I,J)
  166 CONTINUE
      CALL MAT(VOC2,N,N,D4M,N,N,D5M,N,N,YYY,4)
      PART1 = 0.
      PART2 = 0.
      DO 366 I=1,N
      SUM(I) = 0.
  366 CONTINUE
      DO 466 I=1,N
      DO 466 J=1,N
      SUM(I) = SUM(I) +XPMU(J)*D5M(J,I)
  466 CONTINUE
      DO 566 J=1,N
      PART1 = PART1+SUM(J)*XPMU(J)
  566 CONTINUE
      XMMU(1) = ZM(1) -XS(1)
      XMMU(2) = ZM(2)- XS(3)
      XMMU(3)= ZM(3)- XS(5)
      DO 666 I=1,M
      SUM2(I) =0.
      DO 666 J=1,M
      VOCI(I,J) = COVM(I,J)
  666 CONTINUE
      CALL MAT(VOCI,M,M,D3M,M,M,D2M,M,M,XXX,4)
      DO 866 I=1,M
      DO 866 J=1,M
      SUM2(I) = SUM2(I) + XMMU(J)*D2M(J,I)
  866 CONTINUE
      DO 966 J=1,M
      PART2 = PART2 + SUM2(J)*XMMU(J)
  966 CONTINUE
      SD = PART1 + PART2
      GO TO 1300
 1299 CONTINUE
      SD = 10.
 1300 CONTINUE
      RETURN
      END
```

```
ISN

0001          SUBROUTINE SORT(ISEC,ISHIP,IRAD)
        C     SUBROUTINE SORT EXAMINES EACH TRACK IN THE SECTOR UNDER CONSIDERATION.
        C     IF THE TRACK IS A PARTICIPATING MEMBER BIAS ERRORS ARE REDUCED.
        C     IF THE TRACK IS NOT AN OWNSHIP MPT SUBROUTINE TIMCON IS CALLED TO
        C     PREPARE INFO ON ASSOCIATED DETECTION FOR TRANSMISSION OVER THE LINK.
        C     IF THE TRACK IS AN OWNSHIP MPT THE LINKED FILE 'TESTO' IS LOADED
        C     WITH INFO FOR ACCESSING DETECTION FILES. 'TESTO' CONTAINS THE I.D.
        C     NOS.,SHIP NOS. AND RADAR NOS OF ALL DETECTIONS THAT HAVE RECENTLY
        C     BEEN ASSOCIATED WITH NT.  TESTO IS LINKED BY FILID AND FILFX
0002          COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
             1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
0003          COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
0004          COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
             10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
             2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
0005          COMMON/CORAS/DETSX(256),ISTOR(256),DETID(512)
0006          COMMON/PAR2/SLAT(20),SLOG(20),SHT(20),SHD(20)
0007          COMMON/PAR7/ ROLL(20),PITCH(20),RPHASE(20),PPHASE(20)
0008          COMMON/DUMSEC/DUMSX(64,5),DUMID(512,5)
0009          COMMON/NDECK/AZND(20,5,3),ELND(20,5,3),RNND(20,5,3)
0010          COMMON/RANTIM/TRAN(20,3,5)
0011          COMMON/DIST/SD(256)
0012          COMMON/SORT/FILFX(256,5),FILID(512)
0013          DIMENSION RAND(100)
0014          INTEGER DUMSX,DETSX,PTFST,TRKST,TESTO,FILID,FILFX,DUMID
0015          NT = DUMSX(ISEC,ISHIP)
0016       10 CONTINUE
0017          IF(NT.EQ.0) GO TO 999
0018          LOC = DETSX(NT)
0019          IDETNO = ISTOR(LOC)
        C     ARE THERE ANY DETECTIONS ASSOCIATED WITH TRACK NT
0020          IF(IDETNO.EQ.0) GO TO 20
0021          KS = PTFST(NT,ISHIP)
0022          MT = TRKST(NT,ISHIP)
        C MPFLAG = 1 INDICATES PARTICIPATING PLATFORM
0023          IF(MPFLAG(MT,KS).NE.0) GO TO 30
0024          IF(KS.EQ.ISHIP) GO TO 40
0025          ROLLNK(IDETNO,ISHIP) = ROLL(ISHIP)
0026          PITLNK(IDETNO,ISHIP) = PITCH(ISHIP)
0027          SHDLNK(IDETNO,ISHIP) = SHD(ISHIP)
        CCCC  INSERT RANDOM DELAY
0028          NNN = 1
0029          CALL VRANF(RAND,NNN)
0030          TRAN(IDETNO,IRAD,ISHIP) = (RAND(1)*1.9)+.1 +TMS(IDETNO,IRAD,ISHIP)
0031          SDIST = SD(LOC)
0032          CALL TIMCON(IDETNO,MT,KS,ISHIP,IRAD,ISEC,SDIST)
0033          GO TO 20
0034       30 CONTINUE
0035          GO TO 20
0036       40 CONTINUE
0037          IDROP = 1
0038          CALL DETLOC(NT,LOC,IDROP)
0039          TESTO(LOC,1) = IDETNO
0040          TESTO(LOC,2) = ISHIP
0041          TESTO(LOC,3) = IRAD
0042          FILID(LOC) = FILFX(NT,ISHIP)
0043          FILFX(NT,ISHIP) = LOC
0044       20 CONTINUE
0045          NT = DUMID(NT,ISHIP)
0046          GO TO 10
0047      999 CONTINUE
0048          RETURN
0049          END
```

CSN

```
      SUBROUTINE TIMCON(IDETNO,MT,KS,IS,IR,ISEC,SDIST)
      COMMON/MODULO/ISLOT(20,5,60),IKEY(3),IMOD20,IMOD60
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),THRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
     10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
     2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
      COMMON/LOCLNK/LASLNK,FULLNK,LISLNK(20),NEXLNK
      COMMON/RANTIM/TRAN(20,3,5)
      INTEGER FULLNK
      IF(FULLNK.NE.0) GO TO 30
C INITIATE PROCEDURE FOR PURGING FILE
      I = 1
   10 CONTINUE
      ID = LNKSTO(I,1)
      IS = LNKSTO(I,2)
      IR = LNKSTO(I,3)
      IF(TMS(ID,IR,IS).LT.(THRK(ISEC,IR,IS)-30.)) GO TO 20
      CALL LNKDRP(I)
      IDRP = 0
      CALL LNKLOC(I,IDRP)
   20 CONTINUE
      I = I+1
      IF(I.LT.20) GO TO 10
   30 CONTINUE
      PRINT 301,TMS(IDETNO,IR,IS),SDIST,IS,TRAN(IDETNO,IR,IS)
  301 FORMAT(/10X,'DETECT TIME',F10.3,5X,'STATDIST',F10.3,5X,'PLATFORM',
     1I4,5X,'TRANS TIME',F10.3/)
      ITIME = TMS(IDETNO,IR,IS)
      MOD60 = MOD(ITIME,IMOD60)
      IMOD = MOD60 + 1
      IMODM1 = IMOD-1
      IMODM2 = IMOD-2
      IMODM3 = IMOD-3
      IMODM4 = IMOD-4
C IF TIME SLOT IS FULL DETECTION IS NOT TRANSMITTED
      IF(ISLOT(MT,KS,IMODM4).NE.0) GO TO 40
      IF(ISLOT(MT,KS,IMODM3).NE.0) GO TO 40
      IF(ISLOT(MT,KS,IMODM2).NE.0) GO TO 40
      IF(ISLOT(MT,KS,IMODM1).NE.0) GO TO 40
      IF(ISLOT(MT,KS,IMOD).NE.0) GO TO 40
      ISLOT(MT,KS,IMOD) = 1
      IDRP = 1
      CALL LNKLOC(LOC,IDRP)
      LNKID(LOC) = LNKFSX
      LNKFSX = LOC
      LNKSTO(LOC,1) = IDETNO
      LNKSTO(LOC,2) = IS
      LNKSTO(LOC,3) = IR
      LNKSTO(LOC,4) = MT
      LNKSTO(LOC,5) = KS
   40 CONTINUE
      RETURN
      END
```

CSN

```
      SUBROUTINE LNKDET(ISHIP,ISEC,IRAD)
C     SUBROUTINE LNKDET PLACES DETECTIONS FROM THE LINK IN THE TETO FILE
C     FOR EACH TRACK IN ISEC,THE DETECTIONS ARE ORDERED IN TIME AND
C     PLACED IN THE ILOC FILE. LNKDET CALLS UPDATE AFTER ORDERING THE
C     DETECTIONS
      COMMON/RANTIM/TRAN(20,3,5)
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKRIN(20,3,5)
      COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
      COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
     10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
     2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
      COMMON/DUMSEC/DUMSX(64,5),DUMID(512,5)
      COMMON/SORT/FILFX(256,5),FILID(512)
      INTEGER DUMST,TESTO,FILID,FILFX,DUMSX,DUMID,PTFST,TRKST
      NT = DUMSX(ISEC,ISHIP)
      IF(NT.EQ.0) GO TO 999
      LOC = LNKFSX
C     LNKFSX IS THE LOCATION OF FIRST ASSOCIATED DETECTION IN LINK FILE
   10 CONTINUE
C     ALL LINK DETECTIONS CONSIDERED
      IF(LOC.EQ.0) GO TO 150
      KS = LNKSTO(LOC,5)
C     DOES THIS DETECTION CORRELATE WITH ISHIP TRACK?
      IF(KS.NE.ISHIP) GO TO 101
      ID = LNKSTO(LOC,1)
      IS = LNKSTO(LOC,2)
      IR = LNKSTO(LOC,3)
      MT = LNKSTO(LOC,4)
C     IS DETECTION ASSOCIATED WITH PARTICIPATING PLATFORM
       IF(MPFLAG(MT,KS).NE.0) GO TO 100
      NT = DUMST(KS,MT,ISHIP)
      IF(TRAN(ID,IR,IS).GE.TMRK(ISEC,IRAD,ISHIP)) GO TO 101
C     WAS DETECTION MADE BEFORE LAST UPDATE
      IF(TMS(ID,IR,IS).LE. TLAST(NT,ISHIP)) GO TO 100
      IDRP  = 1
C     GET NEW LOCATION FOR LINK DETECTION IN TESTO FILE
      CALL DETLOC(NT,LOCC,IDRP)
      TESTO(LOCC,1) = ID
      TESTO(LOCC,2) = IS
      TESTO(LOCC,3) = IR
      FILID(LOCC) = FILFX(NT,ISHIP)
      FILFX(NT,ISHIP) = LOCC
  100 CONTINUE
      LOCOLD = LOC
      LOC = LNKID(LOC)
      CALL LNKDRP(LOCOLD)
      IDRP = 0
```

LNKDET

```
      CALL LNKLOC(LOCOLD,IDRP)
      GO TO 10
  101 CONTINUE
      LOC = LNKID(LOC)
      GO TO 10
  150 CONTINUE
C     GO THRU TRACKS AGAIN
      NT = DUMSX(ISEC,ISHIP)
  200 CONTINUE
      K = 999
      IF(NT.EQ.0) GO TO 999
      KSHIP = PTFST(NT,ISHIP)
      IF(KSHIP.NE.ISHIP) GO TO 280
      MT = TRKST(NT,ISHIP)
      IF(MPFLAG(MT,KSHIP).EQ.0) GO TO 205
      GO TO 270
  205 CONTINUE
      K = 0
      LOCC = FILFX(NT,ISHIP)
C     ARE THERE NO DETECTIONS ASSOCIATED WITH THIS TRACK
      IF(LOCC.EQ.0) GO TO 280
  210 CONTINUE
      K = K+1
      J = K
  220 CONTINUE
C     FIRST PASS
      IF(J.EQ.1) GO TO 260
C     ORDER COMBINED DETECTIONS IN TIME
      ID = TESTO(LOCC,1)
      IS = TESTO(LOCC,2)
      IR = TESTO(LOCC,3)
      IDD = TESTO(ILOC(J-1),1)
      ISS = TESTO(ILOC(J-1),2)
      IRR = TESTO(ILOC(J-1),3)
      IF(TMS(ID,IR,IS).GT.TMS(IDD,IRR,ISS)) GO TO 260
      ILOC(J) = ILOC(J-1)
      J = J-1
      GO TO 220
  260 CONTINUE
      ILOC(J) = LOCC
      LOCC = FILID(LOCC)
C     HAVE ALL DETECTIONS BEEN ORDERED
      IF(LOCC.NE.0) GO TO 210
  270 CONTINUE
      CALL UPDATE(NT,ISEC,IRAD,ISHIP,K)
  280 CONTINUE
      NT = DUMID(NT,ISHIP)
      GO TO 200
  999 CONTINUE
      RETURN
```

```
CSN

      SUBROUTINE UPDATE(NT,ISEC,IRAD,ISHIP,K)
C     SUBROUTINE UPDATE GOES THRU THE LIST OF ORDERED DETECTIONS WHICH
C     ARE ASSOCIATED WITH TRACK NT AND UPDATES POSITION AND VELOCITY OF
C     NT BASED ON THESE DETECTIONS.  UPDATES ARE MADE TO'TIMUP'WHICH IS
C     CURRENT TIME LESS SOME SPECIFIED TIME LAG
      COMMON/PLOT/IPLT(1000),XX1(1000),YYY(1000),NP,YYN(1000),IYIS(1000)
      COMMON/KAL2/ PS(20,20),PP(20,20),COVS(20,20),COVM(20,20),XP(20)
      COMMON/KAL4/IFIRST(20,5),DIM1,DIM2,DIM3
      COMMON/UPDATE/ILOC(20),TIMLAG,TESTO(512,3),TLAST(256,5)
      COMMON/LINK/LNKFSX,LNKSTO(20,5),DUMST(5,20,5),TRKST(20,20),PTFST(2
     10,20),LNKID(20),TIMLNK(20,5),ROLLNK(20,5),PITLNK(20,5),SHDLNK(20,5
     2),XSMO(20,20,5),COVSMO(10,10,20,5),PRECOV(10,10,20,5),MPFLAG(20,5)
      COMMON/DETECT/XYZMS(20,3,3,5),TMS(20,3,5),COVMS(20,3,3,3,5),
     1LSTBIN(64,100,3,5),XS(20),TMRK(64,3,5),TRATG(20),LNKBIN(20,3,5)
      COMMON/NEW/AZ(20,20),RG(20,20),EL(20,20)
      COMMON/DATEUP/NUMTRG,NUMSHP
      COMMON/TRUE/XYZTRU(20,3,3,5)
      COMMON/DETFIL/IDET(3,5),IDTA(256,3,5)
      INTEGER TESTO,TRKST,PTFST,DIM1,DIM2,DIM3,DUMST,TRATG,MPFLAG
      N= DIM2
      MT = TRKST(NT,ISHIP)
C K=999 INDICATES UPDATE OF PARTICIPATING PLATFORM
      IF(K.EQ. 999) GO TO 80
      TIMUP = TMRK(ISEC,IRAD,ISHIP) - TIMLAG
      DO 55 I=1,N
      XS(I) = XSMO(I,MT,ISHIP)
      DO 55 L=1,N
      PS(I,L) = COVSMO(I,L,MT,ISHIP)
   55 CONTINUE
      J=1
   10 CONTINUE
      ID = TESTO(ILOC(J),1)
      IS = TESTO(ILOC(J),2)
      IR = TESTO(ILOC(J),3)
      IT = IDTA(ID,IR,IS)
C     IS DETECTION TIME GREATER THAN TIMUP.
      IF(TMS(ID,IR,IS).GT.TIMUP) GO TO 20
      X = XYZMS(ID,1,IR,IS)
      Y = XYZMS(ID,2,IR,IS)
      Z = XYZMS(ID,3,IR,IS)
      TDEL = TMS(ID,IR,IS) - TLAST(NT,ISHIP)
      IF(ISHIP.EQ.IS) GO TO 40
      CALL COVLNK(IS,ID,IR,ISHIP)
      XT = X
      YT = Y
      ZT = Z
      XTT = XYZTRU(ID,1,IR,IS)
      YTT = XYZTRU(ID,2,IR,IS)
```

UPDATE

```
      ZTT = XYZTRU(ID,3,IR,IS)
C     NOISY STABILIZED COORDINATES WITH RESPECT TO ISHIP
      CALL TRANSF(XT,YT,ZT,X,Y,Z,ISHIP,IS)
C     TRUE COORDINATES WITH RESPECT ISHIP
      CALL TRANSF(XTT,YTT,ZTT,XTRU,YTRU,ZTRU,ISHIP,IS)
      GO TO 50
   40 CONTINUE
      XTRU = XYZTRU(ID,1,IR,IS)
      YTRU = XYZTRU(ID,2,IR,IS)
      ZTRU = XYZTRU(ID,3,IR,IS)
      DO 45 I=1,3
      DO 45 L=1,3
      COVM(I,L) = COVMS(ID,I,L,IR,IS)
   45 CONTINUE
   50 CONTINUE
      IF(IFIRST(MT,ISHIP).NE.0) GO TO 60
      XP(1) = X
      XP(2) = XSMO(2,MT,ISHIP)
      XP(3) = Y
      XP(4) = XSMO(4,MT,ISHIP)
      XP(5) = Z
      XP(6) = XSMO(6,MT,ISHIP)
   60 CONTINUE
      KFLAG = 1
      CALL KALMAN(TDEL,MT,X,Y,Z,ISHIP,KFLAG,SD)
      IFIRST(MT,ISHIP) = IFIRST(MT,ISHIP) + 1
      IF(MPFLAG(MT,ISHIP).NE.0) GO TO 90
      TLAST(MT,ISHIP) = TMS(ID,IR,IS)
      TIMLNK(MT,ISHIP) = TMS(ID,IR,IS)
      IDRP = 0
      CALL DETLOC(NT,ILOC(J),IDRP)
      CALL DETDRP(NT,ILOC(J),ISHIP)
      PRINT 300,TIMLNK(MT,ISHIP),IT,XTRU,YTRU,ZTRU,IS
  300 FORMAT(/10X,F10.3,5X,'TARGET',I4,3X,'TRUE',2X,3F13.3,43X,I5)
      PRINT 302,TMRK(ISEC,IRAD,ISHIP),X,Y,Z
  302 FORMAT(10X,F10.3,17X,'NOISY',2X,3F13.3)
      PRINT 304,XS(1),XS(3),XS(5),XS(2),XS(4),XS(6)
  304 FORMAT(36X,'SMOOTH',2X,6F13.3//)
      IF(IT.NE.1) GO TO 70
      YYN(NP) = Y
      YYY(NP) = XS(3)
      XX1(NP) = TIMLNK(MT,ISHIP)
      IYIS(NP) = IS
      NP = NP+1
   70 CONTINUE
C     ARE THERE MORE DETECTIONS TO BE CONSIDERED.
      IF(J.EQ.K) GO TO 30
      J=J+1
      GO TO 10
   80 CONTINUE
```

UPDATE

CON

```
C IF IT HAS BEEN MORE THAN 2 SECONDS SINCE THE LAST UPDATE OF A SHIP
C IT WILL BE UPDATED TO WITHIN ONE SECOND OF THE SECTOR CROSSING TIME.
      IF((THRK(ISEC,IRAD,ISHIP)-TLAST(NT,ISHIP)).LT.2) GO TO 999
      TIM1 = THRK(ISEC,IRAD,ISHIP)-1.
      TDEL = TIM1-TLAST(NT,ISHIP)
      IF(TDEL.LT. 1.) GO TO 999
      J = ISHIP
      K = MPFLAG(MT,ISHIP)
      CALL SHPGEN(TIM1,NUMSHP)
      CALL SCOORD(NUMTRG,NUMSHP,J)
      X = RG(K,J)*SIN(AZ(K,J))*COS(EL(K,J))
      Y = RG(K,J)*COS(AZ(K,J))*COS(EL(K,J))
      Z = RG(K,J)*SIN(EL(K,J))
      DO 85 I=1,3
      DO 85 L=1,3
      COVM(I,L) = 0.1
   85 CONTINUE
      COVM(1,1) = 100.
      COVM(2,2) = 100.
      COVM(3,3) = 100.
      GO TO 50
   90 CONTINUE
      TLAST(NT,ISHIP) = TIM1
      TIMLNK(MT,ISHIP) = TIM1
      GO TO 30
   20 CONTINUE
      IF(J.EQ.1) GO TO 999
   30 CONTINUE
      DO 35 I=1,N
      XSMO(I,MT,ISHIP) = XS(I)
      DO 35 J=1,N
      COVSMO(I,J,MT,ISHIP) = PS(I,J)
      PRECOV(I,J,MT,ISHIP) = PP(I,J)
   35 CONTINUE
      IF(MPFLAG(MT,ISHIP).NE.0) GO TO 180
      GO TO 185
  180 CONTINUE
      NSHIP = K-NUMTRG
      PRINT 301,TIMLNK(MT,ISHIP),NSHIP,X,Y,Z,ISHIP
  301 FORMAT(10X,F10.3,7X,'SHIP',I4,3X,'TRUE',2X,3F13.3,43X,I5)
  185 CONTINUE
  999 CONTINUE
      RETURN
      END
```